USER INTERFACE PARADIGMS IN DIGITAL AUDIO WORKSTATIONS

EXAMINING AND MODERNISING ESTABLISHED MODELS

PETRI MYLLYS

User Interface Paradigms in Digital Audio Workstations

Examining and Modernising Established Models

Petri Myllys 2014
Master's final project

Department of Music Technology,
Faculty of Music Education, Jazz and Folk Music,
Sibelius Academy,
University of the Arts Helsinki

Supervisors: Andrew Bentley and Otto Romanowski

SIBELIUS-ACADEMY

Abstract | Projektin kirjallinen työ

Title	**Number of pages**
User Interface Paradigms in Digital Audio Workstations: Examining and Modernising Established Models	124
Author(s)	**Term**
Petri Myllys	Spring 2014
Degree programme	**Study Line**
Musiikkiteknologia	
Department	
Musiikkikasvatuksen, jazzin ja kansanmusiikin osasto	

Abstract

This thesis describes a project examining the status of established user interface paradigms in digital audio workstations. The description proceeds in two stages. Firstly, the interfaces of prominent digital audio workstations are examined, and the fundamental interface structure is abstracted from the observations. Secondly, a modernised user interface concept is proposed.

Technological frameworks and the background of the current digital audio workstation designs provide frames of reference for the examination of user interfaces. An important attribute of this thesis is the standpoint of the present day: the optimality of the established interface paradigms is assessed in connection with modern personal computing technology and today's music production. On this basis, improvements on the established paradigms are framed, and the resulting design is prosed as an abstract, highly scalable interface concept.

The proposed interface concept offers a modernised approach to mixing in digital audio workstations and demonstrates several benefits of re-evaluating the established interface paradigms. Current interfaces are highly analogous to traditional, specific hardware audio devices. This poses inherent restrictions on the flexibility of the interfaces. Discarding some of these analogies allows the design of an up-to-date user interface that offers flexibility and scalability superior to the established approach.

Keywords

digital audio workstation, user interface, personal computer, sequencer, multitrack recorder, mixing console, mixing, interface concept

Other Information

Table of Contents

1 Introduction

Audio production environments are often networks of devices. Whereas in the analogue studio environment these devices were usually specialised, digital technology has made multipurpose machines possible. Whether the context is a large-scale audio production facility or a small project studio, a computer-based digital audio workstation is likely a cornerstone of that environment.

It is not surprising that computer systems have surpassed analogue devices – computers are highly cost-effective and, by comparison to many analogue devices, virtually maintenance-free. Not all analogue devices have disappeared from the scene, however. New outboard audio processors are being introduced and sold, and what is more, many vintage analogue devices no longer manufactured are sought-after and often still considered essentials of large studios.

Since the very first recordings, the recording industry has been tightly related to technological progress. The first recordings were largely technical proofs of concept, but they did initiate the development of more advanced recording technology – and of course, they were astonishing at the time they were made. Other art forms that have emerged from technological inventions, namely film and photography, share a similar history.

The digital environment is fundamentally different from the analogue environment. On one hand, many tasks that are either tedious or impossible in an analogue environment are effortless to execute in digital systems. Many of these tasks are so common in modern audio production – editing, for instance – that imagining working with the all-analogue systems seems very alien from today's point of view. On the other hand, completely different restrictions apply in the digital domain, the clipping behaviour of devices being a prime example.

Indeed, moving to the digital domain expanded the horizons of music production, but the changes have been happening more gradually. To inspect this phenomenon, development in adjacent fields of art and technology are important to consider. During the era of digital audio production and delivery – from approximately 1982[1] to the present – many information technological revolutions have happened. The Internet has transformed the way information is exchanged and knowledge shared, while personal computers have shrunk to fit the pocket. It is evident that today trends can spread faster than ever.

Whereas the analogue recording studio was a network of many specific devices often connected through a large-scale mixing console, this is commonly no longer the case: music production and audio work are today largely based on computer systems. Working completely "in the box" – i.e. using only a minimum amount of external hardware connected to the computer system – is not uncommon. If external devices are used, the hub of the environment is still likely to be the computer-based system – the digital audio workstation.

Computing technology is not a steady and fully developed field. Computer-based digital audio systems have now been the prevalent form of audio workstation for many years, and the paradigm of desktop computing has been the basis for common personal computers. During the history of personal computing certain interfacing conventions became so widespread that they are now ubiquitous, e.g. pointer-based graphical user interfaces used with a mouse and a keyboard. Laptop computers became common in addition to desktop computers, but incorporated primarily the same interaction paradigms.

In recent years, however, the conventions of personal computing have changed vastly. New device categories, such as tablet computers, have become extremely popular. Mobile phones have transformed to "smart phones" that are, in fact,

1 Philips and Sony produced Red Book standards for compact discs in 1980, and in 1982, Philips introduced the first CD player (BBC, 2007).

primarily computers. These new form factors also brought new interaction methods: instead of the traditional input devices, current mobile computers are often interacted more directly using touchscreens.

Digital audio workstations have not been keeping up with the rapid changes in common computing paradigms. Modern digital audio workstation software still largely mimic analogue devices both functionally and visually. The influence of tape recorders, mixing consoles, and outboard effect devices is evident in practically every major digital audio workstation; the interaction is based on a simulation of the analogue environment. Although this is not necessarily useless or detrimental, such analogy does restrict the possibilities specific to the digital system.

The drawbacks of the interface paradigms in digital audio workstations are becoming more and more noticeable. The user base of digital audio workstations is arguably very different from what it was when the first versions of the software were introduced, and people are generally used to different computing paradigms in their everyday lives. This is a challenge many specialised computer-based tools need to address; the power and precision of the established paradigms should not be lost, yet the tool should appeal to different generations of users.

Inspecting the status of the digital audio workstation interfaces in relation to the current technological situation provides the foundation for this thesis; this text is concerned with whether the established user interface paradigms in digital audio workstations are still optimal. The capabilities offered by the present-day personal computing technology, the usability implications of the established interface paradigms, and the needs that emerge from today's music production are important considerations in this text.

This thesis consists of two main aspects: the examination of the established user interface paradigms and the development of an interface concept. The approach used is somewhat different from many other texts that describe the development of a user interface. The purpose of this thesis is not only to offer a written part for an interface concept, but also to describe comprehensively the paradigms that constitute the established digital audio workstation user interface. Therefore, significant emphasis has been placed on the examination of the common interface structure and the background of current digital audio workstations.

The main aim of the proposed interface concept is to offer a modernised approach to mixing in digital audio workstations. The concept describes a block-based

interface that prioritises flexibility and versatility, not forgetting simplicity. The fundamental, abstract interface structure is designed to be only loosely dependent on the characteristics of the input device and the display device, and the concept can therefore be used with a variety of distinct devices.

Another noteworthy attribute of this thesis is the importance of the illustrations. The schematisations of the examined aspects and the carefully designed representations of the proposed interface concept constitute a great portion of the figures of this text. These original illustrations are drawn specifically for the requirements of this thesis. Especially the figures portraying the interface concept are integral to this project, in some ways even more so than the written part.

The discussion is divided into four Chapters. Firstly, the multidisciplinary basis for the paradigms is described in Chapter 2 "Underlying concepts of modern audio workstations and user interfaces". These topics form a reference point for the interface paradigm examination presented in Chapter 3 "Examination of established user interface paradigms" and for the interface concept proposed in Chapter 4 "Processing with blocks – an interface concept for mixing". Lastly, the results of this thesis and an overall view over the topics discussed are presented in Chapter 5 "Conclusions".

2 Underlying concepts of modern audio workstations and user interfaces

Inspecting the interface design in digital audio workstations reveals that – in addition to the concept itself – adjacent fields need to be considered. Both the interface design and digital audio workstations are dependent on the prevailing technological conditions. Moreover, inspecting interfacing paradigms essentially requires an understanding of the underlying technology. This is also crucial in assessing the relevance of such paradigms. Therefore, the point at which digital audio workstations, user interface design, and personal computing intersect is in the locus of attention in this thesis.

Discussing established paradigms is hardly possible without an appropriate inspection of the background. In the case of this text, the user interface paradigms in modern digital audio workstations are essential, and therefore, the emergence of the computer-based audio workstation is discussed. In fact, substantial attention is given to the examination of the background in this thesis for two reasons. Firstly, using a current digital audio workstation quickly reveals that the legacy of the specialised hardware devices is still prominent. Secondly, computer-based systems have largely superseded the original devices, and understanding the current interface structure requires tracing the paradigms.

The context in which digital audio workstations are used is also noteworthy when examining the related conventions. There is a clear interrelationship between music production and other forms of sound design, in terms of both the tools and the procedures. Nevertheless, concepts within these fields are not necessarily interchangeable. In this thesis, audio workstations are inspected from the standpoint of music production usage, but this does not mean findings presented in this text would not be applicable to other forms of sound design. In fact, the boundaries between sound design, composition, and music production are somewhat vague.

There has been some research in the field of computer-based music production systems involving usability and interaction over the last decade. Some of the key concepts of this thesis have been discussed in Matthew Duignan's (2008) relatively recent dissertation *Computer mediated music production: A study of abstraction and activity*. Duignan examines the abstractions present in the music production systems, placing great emphasis on the multitrack-mixing metaphor. Chris Nash's (2011) dissertation *Supporting Virtuosity and Flow in Computer Music* also inspects computer-based music production systems, but emphasises strongly creativity. The concept of creativity is, however, beyond the scope of this thesis.

In addition, the research paper *Metaphors for Electronic Music Production in Reason and Live* by Duignan, et al. (2004) presents an examination of one of the central concepts inspected also in this thesis: the relationship between the user interface metaphors in music production systems and the systems' usability. *A taxonomy of sequencer user-interfaces* by Duignan, Noble, and Biddle (2005) is also relevant, as sequencing has been one of the fundamental tasks for digital audio workstations old and new.

This Chapter presents the essential concepts underlying the modern digital audio workstations, and in addition, relevant usability principles are discussed. The section 2.1 "Terminology" presents key terms relating to the intersecting fields described above. The technological grounds are inspected in the section 2.2 "Brief review of the technological basis", and the usability concepts are considered in the section 2.3 "Concepts of interaction and usability". Lastly, the section 2.4 "Background of computer-based digital audio workstations" provides an overview of the basis for the current, computer-based digital audio workstations.

2.1 Terminology

The main topic of this thesis, the user interface paradigms in digital audio workstations, is a multidisciplinary subject. Consequently, some of the terms central to the discussion of this topic have multiple meanings and interpretations. Terminology essential to the user interfaces of the modern computer-based audio workstations is therefore discussed here. The discussion presents the context in which the terms are used in this thesis; the definitions are restricted to the scope of this text and will not necessarily be completely accurate across other fields.

Digital audio workstation

Today, "digital audio workstation" (DAW) often refers to a multifunctional computer-based audio system offering the means to handle most of the typical audio production tasks. Huber and Runstein (2005: 251–252) describe these systems having functionality for multitrack recording, editing, and mixing, MIDI sequencing, plug-in signal processing, and integrating virtual instruments. The media supports this definition by associating the term with software applications having the functions listed above (SOS Publications Group, 201?; The MusicRadar Team, 2012). However, the computer system running the audio software may also be referred to as a digital audio workstation (Cakewalk, 2013a).

The relationship between the software and the hardware is therefore slightly ambiguous in the term "digital audio workstation". The two are nonetheless inseparable, and a digital audio workstation only exists as a combination of the hardware and the software. This fact is not dismissed in this text; on the contrary, some emphasis is given to the hardware per se when examining changes in personal computer systems and the attributes of touchscreen devices. Nevertheless, this thesis focuses on software paradigms, and in this text, "digital audio workstation" generally refers to the audio software running on typical computer hardware. This approach serves as a guiding principle that is further discussed when necessary, as referring to any computer hardware as "typical" is becoming increasingly difficult.

The term "audio software" is also indefinite, however, and needs to be disambiguated. The categorisation of audio software is partly based on conventions: one of the prominent examples is that audio editing software, e.g. Steinberg WaveLab (Steinberg Media Technologies, 2014), are rarely called digital audio workstations. Based on the history of recording and mixing devices, digital audio workstations

are in this thesis defined as multifunctional software audio production systems with functionality including, but not limited to, recording, editing, sequencing, mixing, and manipulating musical control data in a non-audio format (commonly MIDI, standing for Musical Instrument Digital Interface).

Personal computer

"Personal computing" refers here to the act of using common consumer-grade computing devices ranging from small handheld devices, often also called mobile devices, to desktop computers. Many of the terms used previously to describe specific kinds of personal computers, e.g. microcomputer and home computer, are not in common use anymore whereas "personal computer" – commonly abbreviated to PC – has remained remarkably appropriate during the approximately half a century the term has been in use. Some narrowly defined terms are still in use, however: for example, modern "smart phones" are still called phones even though they resemble more and more PCs. It is thus reasonable to refer to all of these common computing devices as personal computers.

Despite being literally quite accurate in describing the variety of modern computers, "personal computer" does have substantial historical connotations; this suggests a fresh term could be beneficial. In addition, the 'personal' aspect in these devices seems to be increasingly in doubt, which is further discussed in the subsection 2.2.3 "Changes in personal computing paradigms". Nevertheless, "personal computing" is used instead of just "computing" in this thesis to emphasise the user interaction.

Desktop computer

"Desktop computing" and "desktop computer" refer in this text to the traditional paradigm of personal computing with interface devices set physically on a desktop. A variety of different input devices have been developed, e.g. lightpens, joysticks, and graphics tablets (Shneiderman, 1998: 316–323). However, the mouse and the alphanumeric keyboard became universal for common graphical user interface-based desktop computing. A "desktop view" is typically an integral part of the operating systems used in desktop computers, but this view is not the basis for the usage of the term in this text.

Interface

Digital audio workstations commonly feature at least two distinct interfaces, namely the audio interface and the user interface. In addition, audio interfaces often have their own, separate user interfaces. In this thesis, the term "interface" refers to user interfaces. However, Raskin (2000: 2) noted that a user interface is not necessarily graphical, but it is the "way that you accomplish tasks with a product—what you do and how it responds". This is a sensible remark: the definition makes different interfaces – tangible, auditory, graphical, etc. – comparable. Therefore, a user interface in a computer-based system is essentially a means for human–computer interaction.

Gesture

Gesture is a widely used term in user interaction design as well as in music technology. In this text, "gesture" refers to an interaction method. Raskin (2000: 37) defined gesture as "a sequence of human actions completed automatically once set in motion". Within this thesis, however, gestural interaction refers specifically to bodily human–computer interaction.

Mode

Mode is a prevalent term in music, but in this thesis, "mode" refers to a concept of user interface design. Gestures are essential in understanding interface modes; if a given gesture is constantly interpreted in the same way, the system is in a particular mode (Raskin, 2000: 37). Tidwell (2006: 245) remarked that modes can be detrimental if the user is unaware of the currently active mode. However, Tidwell (2006: 245) added that the problem is easily overcome by representing the active mode with, for example, the mouse cursor. Raskin (2000: 42), in fact, provided a double-barrelled definition for modes, which extends to the active state:

> "*A human-machine interface is* modal *with respect to a given gesture when (1) the current state of the interface is not the user's locus of attention and (2) the interface will execute one among several different possible responses to the gesture, depending on the system's current state.*" (italics in the original)

Therefore, according to Raskin (2000: 42), the modality of the interface depends on whether the user is constantly aware of the current system state. In this thesis, describing certain interface functions as modal is based on the simple definition relating to the relationship between gestures and interpretations – regardless of the representation of the modal state.

Channel

Stereo channels and other multichannel entities are in this thesis included in the concept of channel. Channels are discussed from the standpoint of user interface representations, and current digital audio workstation interfaces typically allow treating multiple related monophonic channels as a single multichannel entity. In other words, channel is not defined strictly as a path for the transmission of a single signal; instead, a channel, as defined here, may consist of multiple individual signals.

Plug-in

A plug-in is a way to extend the core functionality of a software. The plug-ins used in digital audio workstations can be divided into two distinct types: (1) instrument plug-ins that can be played or programmed to create new audio material and (2) effect plug-ins that manipulate the signal passed through them or create additional sound according to the audio input. Instrument plug-ins include synthesisers, virtual-instruments, and utilities such as signal generators. Effect plug-ins are typically specialised tools for processing the dynamics or the spectrum of the signal or creating reverberations of various kinds.

Technically, a plug-in is not an integral part of the host software. Several plug-in specifications are common today, some of which are supported in various digital audio workstations. Effect plug-ins are often referred to as "insert effects" or "send effects", depending on their signal chain position: "insert" refers to inserting the effect into the signal chain, while "send" refers to sending the signal to another channel which contains the plug-in.

Effect device

The term "device" is used in this thesis not only to refer to specific physical artefacts, but also to describe software entities based on the concepts of such apparatuses. In other words, software plug-in effects, for example, are occasionally referred to as plug-in devices in this thesis; "effect device" is used in this text instead of "effect plug-in" when it is unnecessary to restrict the discussion explicitly to effects implemented as plug-ins.

2.2 Brief review of the technological basis

This section covers briefly the essential technological basis for the modern digital audio workstations. Technical details and signal processing theory are kept to a minimum, as these areas are not in the focus of this thesis. In spite of that, the fundamental way the audio is handled in digital systems is discussed briefly in the subsection 2.2.1 "The audio signal in the digital domain", as this provides the basis for many ubiquitous visual audio representations in computer-based systems.

The subsection 2.2.2 "Digital media and the concept of referencing" describes the effects of the digital media, one of the most central concepts that enabled the development of the digital audio workstation. The recent, radical changes in personal computing – and some prospects – are discussed in the subsection 2.2.3 "Changes in personal computing paradigms". These circumstances are tightly related to the position of the traditional desktop-based digital audio workstation.

2.2.1 The audio signal in the digital domain

Deriving discrete-time signals from continuous-time signals by periodic sampling is common. The rate the samples are taken is referred to as *sampling frequency* or *sampling rate*. In order to avoid aliasing, i.e. reflecting high-frequency signal components into the false frequency range, the sampling frequency needs to be high enough. (Oppenheim and Schafer, 1975: 26–30) According to the *sampling theorem*, a signal containing frequencies up to $r/2$ hertz needs to be sampled at least at a rate of r samples per second in order to represent the signal properly. The frequency $r/2$ is commonly called the *Nyquist frequency*. (Rossing, Moore, and Wheeler, 2002: 482–483)

The binary number system is essential in computing. Historically, the unit of information was not fixed: other systems, such as decimal system, were also used (Buchholz, 1962: 42–44). "Bit", a term invented by J. W. Tukey from the binary digit (Shannon, 1948), can be considered in a number of different ways, but using 0 and 1 to represent the bit states is ubiquitous. The number of different possible messages doubles for each added bit; thus, N bits offer 2^N possibilities.

Sampling signals involves a quantising process, in which representative numbers are assigned to sampled values. Digital signals inherently include quantisation error; samples are quantised to the closest possible number representation, which results in maximum quantisation error of one-half of the size of a quantisation region. The quantisation region size is determined by the number of bits used per sample. (Rossing, Moore, and Wheeler, 2002: 483–484)

The number of bits that represent the sampled values determines the concept commonly referred to as *bit depth*. For N bits, signal-to-quantisation error noise ratio (SQNR) is approximately $6N$ decibels, which results in about 96 decibels of SQNR in 16-bit analogue-to-digital (ADC) converters (Rossing, Moore, and Wheeler, 2002: 484). Similarly, the 24-bit resolution commonly used today results in roughly 144 decibels of theoretical dynamic range. However, device performance restricts the actual dynamic range to approximately 130 dB when using highly sophisticated ADCs (Lavry Engineering, 2012), and to some 115 dB with more affordable devices (PreSonus Audio Electronics, 2013).

Other way to inspect the dynamic range is to consider the signal representations in terms of precision. Computers use operands of varying type to carry out operations. Commonly used operand types include integer, single-precisions floating point, and double-precision floating point (Patterson and Hennessy, 1996: 85). Floating-point arithmetic is ubiquitous in computing (Goldberg, 1991: 5), and in addition, capable of representing a great range with a limited number of bits (IEEE Computer Society, 2008). Therefore, the internal resolution for audio processing in modern computers may often be even higher than in specialised audio hardware devices.

2.2.2 Digital media and the concept of referencing

Magnetic tape used to be the universal medium for sound storage (Huber and Runstein, 2005: 187). Analogue recording to magnetic tape formed a direct relationship between the recorded audio and the tape position. Consequently, everything existed "only once": making a copy of the recording involved re-recording the audio to another tape – an unpractical process that inherently degraded the quality of the audio.

Digital domain removed this restriction: storage media in digital systems, e.g. hard disk drives and digital audio tapes, enabled copying the recording without the loss of quality. Furthermore, computers introduced an interface suitable for exploiting this technology. The fundamental change was that referencing the recorded material became possible; using the unique recording directly was no longer necessary.

Early digital audio workstations ran on computers underpowered for serious audio work. Some manufacturers compensated this with proprietary hardware–software combinations (see the section 2.4 "Background of computer-based digital audio workstations"). The hard disk drive performance has nevertheless been a persistent problem in multitrack audio production. A single hard disk drive is only capable of delivering a certain level of performance – especially without adverse effects, namely excessive heat production and operating noise. Therefore, the performance level of the storage media can still be a bottleneck in digital audio workstations, for example, when working with large sample libraries.

The media performance can be improved by distributing the data over multiple disks using a disk array (Chen, et al., 1994: 150–153). Solid-state drives (SSD) provide another possibility to improve the performance, although SSDs are not necessarily superior to hard disk drives in every situation. In their study on SSDs, Chen, Koufaty, and Zhang (2009: 190–191) reported highly improved performance in random read operations compared to hard disk drives, but found also problems, e.g. performance degradation due to internal fragmentation in higher-end SSDs and poor random write performance in low-end drives. However, solid-state drive technology appears to be advancing rapidly.

2.2.3 Changes in personal computing paradigms

Appreciating the computer-based digital audio workstations is easy when they are considered in relation to the time and circumstances of their inception. That time is, however, approximately a quarter century ago, after which the frontiers of technology have moved vastly. Many of the original factors restricting DAWs, e.g. bottlenecks in storage media performance and access, have diminished or disappeared altogether. Computing power is well ahead of what is required for many of the traditional music production tasks.

It is from these kinds of advancements that completely new issues arise. According to Gartner's (2013) recent prediction, the number of desktop and notebook computers shipped worldwide is going to decline in the near future, whereas more portable computers, such as tablets, will increase in quantity. Even disregarding predictions altogether, it is easy to see how immensely popular for instance Apple's iPad product range has become. Remarkably, the iPad was released only a few years ago, in 2010 (Apple, 2010).

The popularity of small-sized touchscreen devices is not surprising: they offer much more intimate user–content connection compared to the traditional combination of a separate input device and a display device. A self-contained, lightweight touchscreen device is also extremely portable. Computing is therefore not tied anymore to the traditional paradigm of desktop devices used with a mouse and a keyboard. The paradigms of digital audio workstations are discussed separately in relation to touchscreen devices in the section 3.5 "Attributes of touchscreen devices".

Touch-based user interfaces are not the only contenders for traditional PCs. Microsoft has already demonstrated the success of gestural input devices with its product Kinect (Walker, 2012), and new gestural products have been released recently – Leap Motion (Leap Motion, 2013) and the new version of Kinect bundled with Microsoft's Xbox One entertainment system (Microsoft, 2013a) being two prominent ones. In addition, Thalmic Labs has announced MYO (Thalmic Labs, 2013), yet another gestural device. However, gestural interfaces do have some disadvantages, such as increased fatigue in comparison to the mouse (Cabral, Morimoto, and Zuffo, 2005; Farhadi-Niaki, GhasemAghaei, and Arya, 2012).

The developmental tendency reveals desktop computing is in an unsustainable state. This is not to say desktop setups would inevitably become useless; on the

contrary, for anything else than short-term use, the traditional PC user ergonomics are still often considered superior to other available options – although the desktop computer arguably also lacks in ergonomics. Nevertheless, the precision and the speed offered by the combination of a mouse and a keyboard still make the desktop computer a sensible choice for many tasks.

Input devices aside, simplifying the human–computer interaction seems to be an important consideration in current personal computing in general. A minimalistic, flat visual style appears to be a current trend. The recent versions of all three major mobile operating systems, namely Apple iOS (Apple, 2013a), Google Android (Google, 2013a), and Microsoft Windows Phone (Microsoft, 2013b), are demonstrative examples of the phenomenon. On one hand, this seems very natural from the standpoint of interaction: current mainstream touchscreens do not offer tactile feedback to support user interface elements analogous to physical world. On the other hand, flat visual style has also received critique: recognising buttons and other actionable objects has caused difficulties (Nielsen, 2012).

Recent desktop systems show similar inclination. In addition to aspects related to visual style, both Microsoft and Apple are also trying to simplify some of the long-standing conventions, such as file management. Recently, Apple added a file tagging system for easier item grouping and searching (Apple, 2013b). Microsoft, on the other hand, is promoting its cloud storage solution SkyDrive[2] as an integral feature of Windows 8 (Microsoft, 2013c). These developments hint that there may be a tendency to replace the literal representation of computer directories with groupings more meaningful for the user.

Modern computers are on one hand becoming more private: personal smart phones are used for many tasks which previously required bigger, possibly shared, devices. At the same time, however, computers are becoming more terminal-like. For example, recent tablet versions of Android operating system include multi-user support (Google, 2013b), in addition to the cloud-syncing services already offered, and Windows 8 synchronises user preferences across different devices (Microsoft, 2013d).

Digital audio workstations have experienced very little innovation while all this technological development has been going on. Fundamental user interaction

2 Microsoft announced in January 2014 that SkyDrive will be renamed OneDrive (Gavin, 2014).

paradigms in the prominent DAWs are still essentially based on imitations of analogue devices on screen.

Recent multitouch devices, i.e. touchscreen devices capable of recognising and following multiple separate touch points, can simulate the analogue mixing consoles in ways the mouse and the keyboard have never managed to – for example, by allowing the simultaneous but individual control of multiple faders. Ironically, however, it is possible that a current user is no longer thoroughly aware of the original, analogue device-based paradigm. Duignan, et al. (2004: 118) raised a similar concern in their study on user-interface metaphors in music production systems already a decade ago:

> "An important question that must be answered to validate the dependence on music hardware metaphors is: what proportion of new and potential users have prior experience with music hardware?"

2.3 Concepts of interaction and usability

This section outlines the theoretical framework used in this thesis for the examination of the digital audio workstation interface paradigms, presented in Chapter 3 "Examination of established user interface paradigms", and for the development of the interface structure proposed in Chapter 4 "Processing with blocks – an interface concept for mixing". Studies on usability and interaction in relation to digital audio workstations are scarce, but other fields offer applicable research.

A host of texts on human–computer interaction, software usability, and user interfaces have been written – these offer viewpoints that can be used to examine paradigms in the digital audio workstations. The concepts presented in this section originate largely from studies on usability and human–centred design. In addition, a feature of interaction design is employed to expand the framework conceptually. Preece, Rogers, and Sharp (2002: 8) described interaction design as "fundamental to all disciplines, fields, and approaches that are concerned with researching and designing computer-based systems for people". Although the focus in this thesis is on interface design, a similar non-restrictive approach is used to contextualise the examination, as the subjects inspected in this text relate to many adjacent fields.

Elaborate discussion on the multitude of concepts contained within the various fields of interaction design and human–computer interaction goes beyond the scope of this thesis. Consequently, extensive consideration of human factors and recent directions in human–computer interaction research, among other things, are excluded from the theoretical framework of this text. Instead, concepts and suggestions presented in the prominent textbooks, e.g. by Nielsen (1993), Norman (1998), and Raskin (2000), are considered and contrasted with differing views introduced in some research.

The subsection 2.3.1 "Perception" outlines concepts relating to human perception, namely the Gestalt laws. The subsection 2.3.2 "Analogies, mappings, metaphors, and affordance" describes concepts that connect the abstract with the concrete in user interface design, and the subsection 2.3.3 "Norman's model of interaction" outlines one way to describe task execution.

2.3.1 Perception

A great number of universal – or highly common – phenomena affect how humans interact. This is not only true for interaction between human beings but also for human–machine interaction. Consequently, studies on human perception provide essential concepts for interface design.

Intuitiveness is often considered a good quality in a user interface, but the features that make an interface seem intuitive are not self-evident. Raskin (2000: 150) argues against calling interfaces intuitive or natural. Raskin states that in reality by "intuitive" users mean the interface operates in a familiar way or is habitual. "Natural" interface feature is, according to Raskin, something that can be used without any instructions. Therefore, although successful interaction and interface designs are often called intuitive, the actual reason why they are highly "usable" may be a different one. Nevertheless, making use of known human perception phenomena helps design an interface that users will perceive similarly.

Gestalt principles

Gestalt psychology, which originated in early 20th century Germany, was based on a notion that the whole differs from the sum of its parts (Rock and Palmer, 1990: 84). In fact, this sentence summarises quite well many of the Gestalt concepts.

One of the central ideas in Gestalt theory was that the human perception is based on the concept of organisation (Rock and Palmer, 1990: 84–85). Gestaltists found that the ability to perceive separate objects was not based only on the retinal image, and the Gestalt laws of grouping were proposed as explanations for the object perception (Rock and Palmer, 1990: 85). *Prägnanz,* another key principle in the concept of organisation, states that the perception of ambiguous images is as simple as the information available allows (Rock and Palmer, 1990: 88).

The Gestalt principles have remained important in many fields, including user interface design. The following laws are included in the Gestalt laws of grouping: proximity, similarity, continuity, familiarity, good shape, common fate, connectedness, and closure (Sinkkonen, et al., 2006: 89–91). Some of the laws are very directly applicable to user interfaces: for example, proximity, similarity, and closure can easily be used for visualising entities. Grouping interface elements according to the Gestalt laws is considered good practice, and unintentional groupings of unrelated elements should be avoided (Nielsen, 1993: 117–118; Sinkkonen, et al., 2006: 91).

2.3.2 Analogies, mappings, metaphors, and affordance

Analogies and metaphors have a close connection in user interfaces – the terms might even be interchangeable in some contexts. The use of mappings and metaphors is strongly encouraged in traditional usability literature. Norman (1998: 23) suggests using natural mappings, such as physical analogies and cultural properties. One of the principles of usability heuristics described by Nielsen (1993: 123) is that "the terminology in user interfaces should be based on the users' language and not on system-oriented terms". In addition to the literal sense of this suggestion, Nielsen (1993: 126–128) emphasises the use of well thought out mappings and metaphors between the user's conceptual model and the information provided by the computer.

In digital audio workstations, parallels are drawn between specific analogue devices and aspects of the computer-based user interface, e.g. between an analogue mixing console and a digital software mixer. Thus, there is often an *analogy* between a function in a DAW and the procedure required to achieve similar results in the analogue environment. These analogies form *metaphors* that act as "concrete handles" for abstract operations. For example, instead of manipulating a channel's 'gain' directly – a task immensely difficult to imagine in a graphical user

interface – user manipulates a *metaphor* of gain, quite commonly a slider representing an object familiar from the analogue mixing consoles.

Analogies to hardware audio devices are typical in digital audio workstations, but their effectiveness should be questioned. In fact, the advocation of analogies and metaphors in user interfaces has not been unanimous. Halasz and Moran (1982: 383) argued against the use of analogies already over 30 years ago:

> "While analogies may ease the way, they are not the most effective way to teach new users. In fact, analogical models can often act as barriers preventing new users from developing an effective understanding of systems."

More recently, Khoury and Simoff (2003) described the restrictiveness of "concrete metaphors", i.e. metaphors based on familiar physical artefacts, in computing environments. Furthermore, the failure of notoriously metaphorical user interface designs has increased the controversiality of metaphors (Blackwell, 2006: 491–493).

The concept of affordance has established a secure position in user interaction discourse. Gibson (1986) initially used the term "affordance" in reference to all the possible actions that an object or an environment offers. Norman (1998: 9) defined affordance with regard to usability as "the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used". Norman (1998: 87–92) also described how affordances of physical objects provide tangible clues, e.g. certain types of handles and grips "afford" instinctive use of the object, while other designs fail to communicate how the object should be operated.

2.3.3 Norman's model of interaction

Donald A. Norman's seven-staged approximate model describes how people perform tasks. In Norman's model, a goal is something to be achieved, intention is a specific action to achieve it, and action is divided into two aspects: execution and evaluation. Initially, the goal is formed. Three consecutive stages – forming the intention, specifying an action, and executing the action – form the execution aspect. The last three stages – perceiving the state of the world, interpreting it, and evaluating the outcome – form the evaluation. (Norman, 1998: 45–49)

With tasks that are more complex, the seven stages of action may form a loop. Unless the initial intention is accurate enough to offer a way to accomplish the task, a new intention needs to be formed, after which the stages are advanced with the newly formed intention. A poor system may cause unnecessary iterations through the chain of stages or halt the advancement from one stage to the next. This phenomenon is often present in systems that incorporate overly complex user interfaces or use interaction methods that are not familiar for the user. In other words, systems that do not match user's expectations are difficult to interpret.

The mismatches between mental states and physical states are referred to as gulfs in Norman's model, the size of the gulf representing the amount of mismatch. The gulf of execution portrays how well the provided system actions match the person's intentions. The gulf of evaluation describes how difficult the results of these intentions are to inspect and interpret. (Norman, 1998: 49–52) A number of things can create such gulfs, e.g. an overly slow system response, an unclear layout of interface elements, or an input method that is neither evident nor demonstrated.

2.4 Background of computer-based digital audio workstations

The workflow based on digital computing technology is relatively recent in audio production industry. This means that digital audio workstations were not initially de facto tools as they are today – instead, they were contenders. This bond to the industry history is an important aspect when inspecting the interaction in audio workstations: it is one of the fundamental reasons behind many interface design conventions still common in the recent versions of the prominent digital audio workstations.

The origins of digital audio workstations can be viewed from several standpoints. According to Nash (2011: 16), "Modern *digital audio workstations (DAWs)* evolved from MIDI sequencer software" (italics in the original). This is a reasonable view, as the sequencing methods established by MIDI sequencers are still prominent. Many of the features in current digital audio workstations, however, do not originate from sequencers; regarding the MIDI sequencer – or even the concept of sequencing in general – as the single origin of DAWs understates many essential aspects.

Instead, considering the modern digital audio workstation as a sum of its parts (if not greater) offers a wider perspective, which portrays the background of the current DAW paradigms more accurately. Duignan (2008: 13–16) described digital audio workstations arising from the combination of several advantages that enabled the computer-based system to become the centrepiece of the studio, these advantages including analogue-to-digital and digital-to-analogue conversion, digital signal processing, and MIDI. On the other hand, the computer proved to be the revolution that enabled combining the essential functions from the preceding devices and the specialised tools. As personal computers became more powerful and capable, these new possibilities were rapidly used also for music production.

In this thesis, the focus is on the user interaction paradigms in modern digital audio workstations. On this basis, the background is divided here into three main categories:

1. sequencers;
2. multitrack recorders;
3. analogue mixing consoles.

This division is partly arbitrary; for example, the development of sequencers, samplers, and recording devices overlapped, and similar features were available in very different products. Nevertheless, these stereotypes help comprehend the interface conventions of current digital audio workstations. The three device categories described above are discussed in the subsections 2.4.1 "Sequencers", 2.4.2 "Multitrack recorders", and 2.4.3 "Analogue mixing consoles".

2.4.1 Sequencers

Digital audio workstations are sometimes referred to as sequencers, but there is a noticeable semantic difference between the two terms. "Sequencer" makes a clear connection to 'sequence', a concept quite common in music, although interpreting the term loosely is in fact more sensible. Regarding "sequence" as an ordered list of relating events describes quite well the fundamental idea of a sequencer. Contrasting "sequencer" with "digital audio workstation" shows that while the latter term is very generic, it explicitly includes audio.

Sequencers originated as hardware devices operated without any computer-based graphical user interfaces. Nevertheless, the fundamental sequencing functions – excluding the interface – were introduced early on. For example, Electronic Music Studios SYNTHI Sequencer 256 was released in 1971 (Vintage Synth Explorer, 2011); the device was, however, capable of recording and reproducing multiple separate sequences of control voltages, and therefore offered basic "multitrack" operation already at that time (Electronic Music Studios, 197?).

During the 1970s and the 1980s, before the personal computing technology offered sufficient performance for music production systems, other solutions were developed for multipurpose audio work. The Synclavier, an all-digital synthesiser, was first produced as a prototype at Dartmouth College in 1975, and the New England Digital Corporation was founded to manufacture the product in 1976 (Manning, 1993: 258). The Fairlight CMI, released a few years later, became another notable synthesiser of that time and a major rival for the Synclavier (Manning, 1993: 260).

These synthesisers offered a considerable performance edge over software synthesis systems of that time: software solutions were non-real-time, whereas the new self-contained hardware synthesisers were able to produce a wide range of sounds in real-time. The synthesis in the Synclavier was based on frequency modulation and additive synthesis, in contrast to Fairlight CMI which used sampling technology. (Manning, 1993: 258–260) This allowed the Fairlight to make use of sounds not originating from the synthesis engine per se. However, in 1986, New England Digital Corporation introduced a direct-to-disk multitrack recorder, which enabled the Synclavier to function as a digital recording system in addition to its synthesis capabilities (Manning, 1993: 260).

From the standpoint of sequencing, the successor of Fairlight CMI proved to be truly remarkable. Fairlight CMI Series II, released c. 1982, featured a view called Page R – the Real-Time Composer (Holmes, 2010: sec. 2, para. 8; sec. 4, para. 4). This view – a graphical sequencing user interface – was evidently astonishing at that time. Page R allowed entering up to 255 different one-bar patterns, each of which consisted of eight separate sequences of musical notes (Carlos and Stewart, 1983: 1). A reproduction of Page R is shown in Figure 1 (Holmes, 1997).

One of the most significant novelties in the 1980s was MIDI, that is to say, Musical Instrument Digital Interface. The MIDI 1.0 Specification was written in 1982 (MIDI Manufacturers Association, 200?). Soon after this, MIDI sequencers

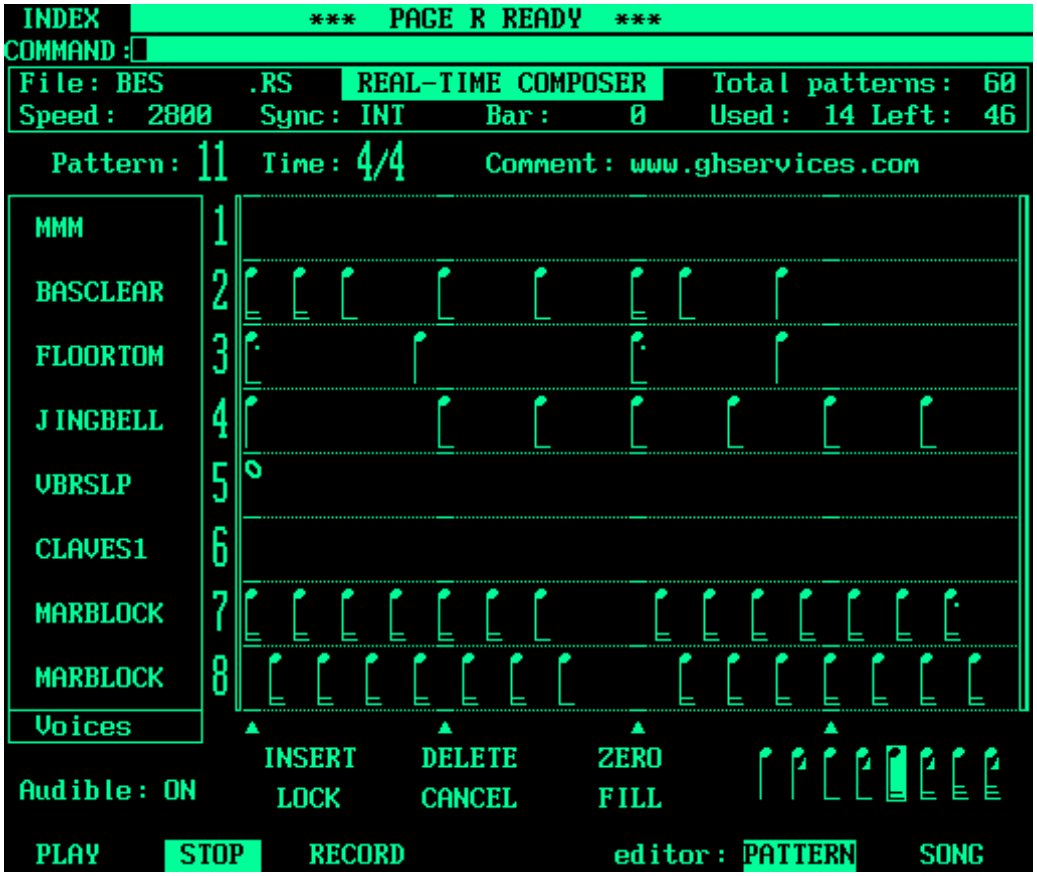


Figure 1: A reproduction of Page R Real-Time Composer view in Fairlight CMI Series II (Holmes, 1997). Used with permission of Greg Holmes.

became available, offering musical data recording capabilities, but not audio recording (White, 2000: 23).

Let us examine a case which demonstrates the development of MIDI sequencers. Pro-24, the predecessor of Cubase, was one of the notable early MIDI sequencers (Manning, 1993: 335). Cubase, the more recent of these two software, was still initially a MIDI sequencer without any audio support per se[3], but Steinberg developed it into an increasingly comprehensive music production package in its successive versions (Steinberg Media Technologies, 2013a). Today, Cubase is arguably one of the most prominent and feature-rich digital audio workstations, and includes all the features commonly associated with DAWs.

A key feature in MIDI sequencers was the graphical track-based timeline user interface, which offered an elegant representation of the sequenced material. This interface has proven to be one of the most significant innovations in the history of

3 Stating there was no 'audio' involved would be misleading, as usually MIDI data eventually generated audio.

digital audio workstations, and should not be trivialised. However, as discussed, the concept of graphical sequencing interface had in fact already been demonstrated in Fairlight CMI Series II.

MIDI no doubt played a profound role in the history of computer music, but it has also been hindering the development of audio workstations increasingly. Moore (1988) reported limitations in transmission rate, bandwidth, and timing in MIDI specification already in the 1980s. These issues have largely remained unrectified. The emergence of alternatives, namely Open Sound Control, accentuates the problems of MIDI. Open Sound Control utilises modern networking technology to communicate between multimedia devices (Wright, 2005). Nevertheless, MIDI is still quite capable of handling some useful data transfers, and discussing the problems any further goes beyond the scope of this text. Moreover, it is crucial to note that although MIDI and the track-based sequencing interface are often used together, the same conceptual user interface paradigm could be used with other forms of control data.

The history of sequencers has not been only about refining a single tool, however. Duignan, Noble, and Biddle (2005) divided sequencing software into four groups: textual language music tools, sample and loop triggers, music visual programming tools, and linear sequencers. This classification suggests that the concept of sequencing has been approached from multiple different standpoints. There are also sequencers that seem to have qualities related to several of these categories – trackers, for instance. Trackers combine, in essence, the principle of timeline sequencing with the concept of text-based notation. For a comprehensive study on trackers, see Nash (2011).

2.4.2 Multitrack recorders

Before the development of digital recording systems, analogue tape recorders dominated the recording industry. Professional analogue multitrack recorders ranged from 2-track to 24-track formats, and typically offered separate heads for erasing, recording, and reproduction (Huber and Runstein, 2005: 190, 194). However, many tape recorders did use a single head for both recording and playing back (Rossing, Moore, and Wheeler, 2002: 499).

The magnetic tape passed the heads in succession, and the recording was fundamentally based on the actions of the heads: the erase head demagnetised the tape,

the record head generated a magnetic field which left a pattern to the magnetic remanence of the tape, and the playback head read the pattern and generated an output voltage. (Rossing, Moore, and Wheeler, 2002: 499)

Common user controls included play, stop, fast forward, rewind, and record buttons, which were used for controlling the transport (Huber and Runstein, 2005: 191). In addition to the basic functionality, additional features such as microprocessor-controlled transport were incorporated for easier operation (Huber and Runstein, 2005: 193–194).

Computer-based hard disk recorders emerged as modernised replacements for completely analogue or digitally controlled analogue recording chains in the early days of relatively affordable personal computing. These recording systems needed to deliver performance comparable to established solutions with preferably some assets that were impossible to replicate in the competing analogue device-based environment. Specialised processing hardware appeared to be the key to make the early computer-based recording systems viable choices. Studer/Editech Dyaxis was one of such systems (Manning, 1993: 338). However, inspecting another comparable product is more sensible from the perspective of the present day.

A company called Digidesign released a suite of Macintosh software in the 1980s, including Sound Designer in 1985. Sound Designer was originally an editing system for the E-mu Emulator II sampler, supporting later also other sampler models. Digidesign's software suite stimulated the development of the Sound Accelerator card, a hardware digital signal processor, which appeared in 1988. (Manning, 1993: 338–339)

Sound Tools, an integrated system introduced in 1989 for Apple Macintosh and in 1990 for Atari ST, combined the Sound Accelerator card with Sound Designer II software (Manning, 1993: 339). A successive system called Digidesign Pro Tools was released in 1991, and quickly became a widespread choice in the audio industry, as it offered capabilities and price-point unmatched by the rival hardware-based products (Robjohns, 2010).

A great success for Digidesign – in addition to overcoming its digital competitors – was attaining a significant position in the field previously almost completely based on analogue systems. The key to Digidesign's success appears to have been the combination of the software-based user interface and the proprietary digital signal processor. Interestingly, this hybrid system is still the basis of the current

version of Pro Tools (now a product of Avid[4]) despite the technological development that has provided central processing units and storage devices capable of delivering high audio performance. However, Avid does offer also software-only versions of Pro Tools today (Avid Technology, 2013b).

A key interface design principle for these new systems was mimicking the paradigms of the analogue devices to realise the domain change. The concept of smoothing the change when substituting a paradigm with another is not, however, specific to audio technology. The history of QWERTY keyboard demonstrates the effects of paradigm popularity: the keyboard layout originated in typewriters, yet it is still extremely popular.

David (1985) shows in his paper on the history of QWERTY that a technique may achieve a dominant position regardless of the quality of its rivals. Network effect is a term used to describe a phenomenon, in which product's value depends on how many other people are using it. This is linked to positive feedback, an important concept in the economics of information technology. A particular product that is popular because of the network of other users creates even more demand, and thus makes the product more and more appealing. (Shapiro and Varian, 1999: 44–46, 173–179)

This pattern offers a plausible explanation for the de facto mixing console view in digital audio workstations; offering a familiar and popular paradigm is economically reasonable. Avid Pro Tools, the DAW of choice in many studios small and large, is a demonstrative example of the positive feedback. Avid offers multiple versions of Pro Tools, ranging from affordable software-only packages to full-featured hardware–software hybrids. When users participate in the network of Pro Tools, they also benefit from the synergy between different studios: session files are directly transferable, and moving between different-sized production houses offers practicality in addition to saving in project costs.

The multitrack tape recorder did not have strong rivals before the computer, and in a way, DAWs were just modern tape recorders working on a different medium. The established track-based paradigm was shifted to a domain that was in many ways superior, thus allowing the DAW to beat the tape recorder directly. The tape recorder, while being a mechanical marvel, was a very clumsy device for anything else than straightforward recording or playback. One of the most severe

4 Avid acquired Digidesign in 1995 (Avid Technology, 2013a).

limitations of the analogue tape becomes evident when examining the editing procedures: whereas stereo-tape could be edited by cutting and splicing, editing multitrack tape was practically impossible.

2.4.3 Analogue mixing consoles

The possibility to record and play multiple tracks back simultaneously generated a need for a means to mix the material, i.e. to control the dynamic and spectral balance of the recorded audio. The tool developed for this task was the analogue mixing console. Solid State Logic 4040 G, one example of the famous large-scale consoles, is shown in Figure 2 (Jussila and Finnvox Studiot, 2013).

Figure 2: Solid State Logic 4040 G large-scale analogue mixing console (Jussila and Finnvox Studiot, 2013). Photograph: Mika Jussila, Finnvox Studiot. Used with permission of Mika Jussila and Finnvox Studiot.

Ultimately, virtually all mixing consoles shared the same basic concept, but factors such as the number of channels, the component quality, the routing and processing possibilities, and differences in the ergonomics and the general design approach differentiated some consoles from their rivals.

In general, a mixing console provided a control point for the user to determine how the sources were combined. The most fundamental purpose of a mixing console was therefore to offer means to adjust different tracks in relation to each other. In practice, mixing consoles included multiple "channel strips" through which inputted signals were routed to sum channels. Mixing consoles allowed monitoring the signals aurally by listening to the channel sums or the individual channels, and visually by inspecting the channel level meters. This concept is illustrated in Figure 3.

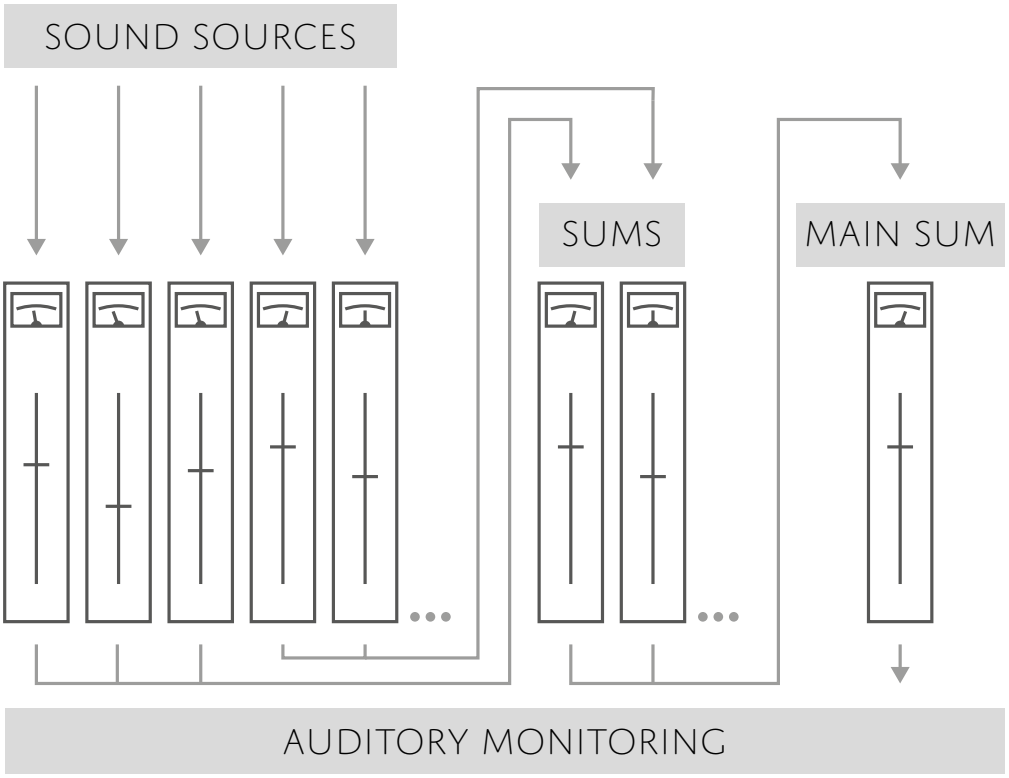


Figure 3: An abstraction of the fundamental analogue mixing console paradigm illustrating "submixing" and summing of the resulting groups. The Figure portrays a situation where the main sum channel is monitored.

An input signal was commonly passed on to a channel equaliser after the initial input gain stage, and after the equaliser, the signal arrived to the channel fader (Robjohns, 1997: sec. 2, para. 2). Auxiliary outputs provided a possibility to "send" signals to other channels in addition to the main signal flow. Typically, auxiliary

outputs were on the signal path immediately before or after the channel fader (Robjohns, 1997: sec. 2, para. 2).

Lastly, the signals were routed either directly to the outputs or to groups that made managing a large number of signals easier and allowed processing multiple signals simultaneously (Robjohns, 1997: sec. 2, para. 3). The majority of mixing consoles offered also insert points that allowed using outboard audio processors in the signal path – either pre-equaliser, post-equaliser but pre-fader, or post-fader (Robjohns, 1997: sec. 10, para. 1). An abstraction of the described channel structure is shown in Figure 4. Additionally, dynamics processing was available in some consoles.

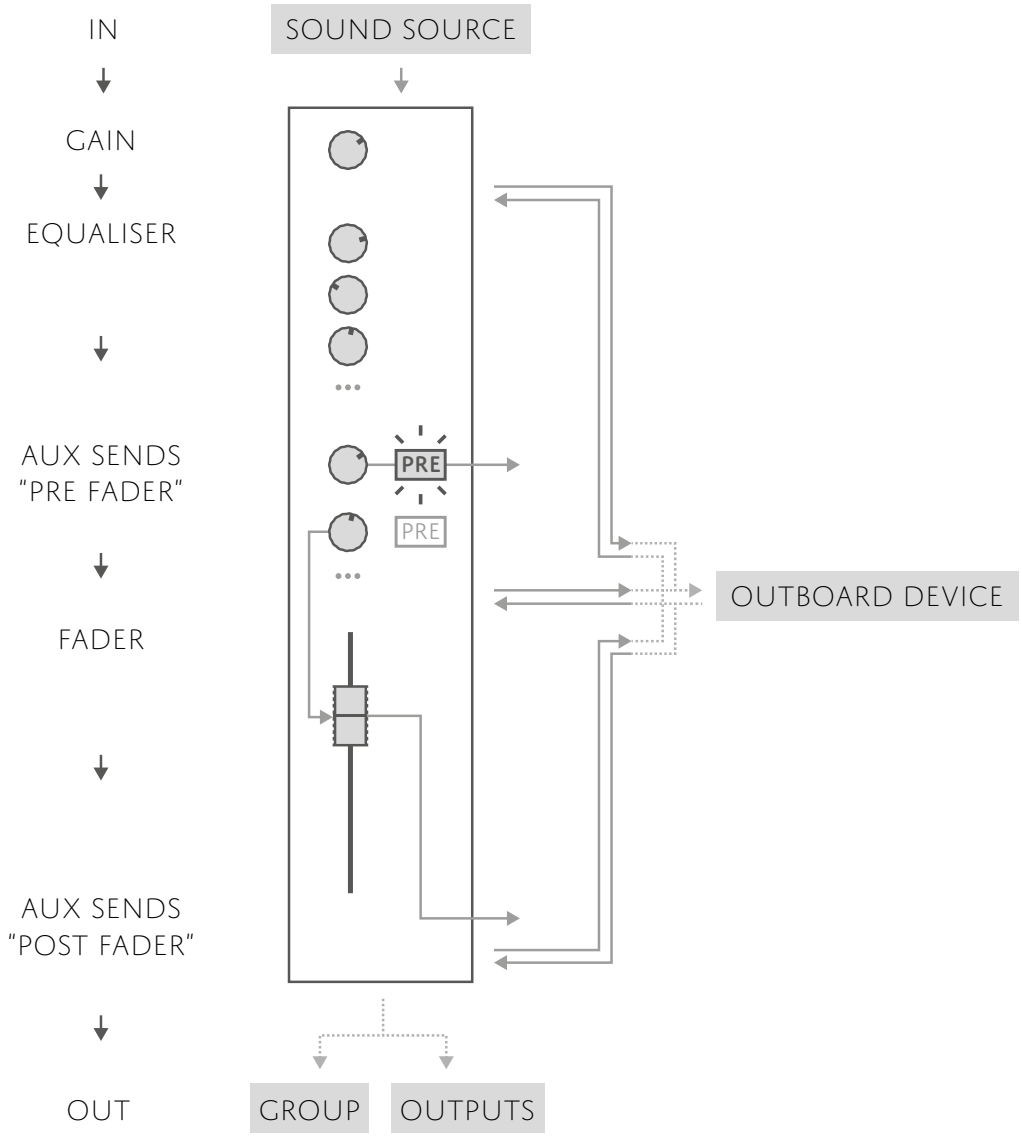


Figure 4: An abstraction of the channel signal path in an analogue mixing console.

Analogue mixing consoles were divided into two categories – split and in-line – based on the design approach. Split consoles had a separate monitor section whereas in-line consoles included the monitoring functionality in the channel strips. Split design did have the advantage of being simple to understand, but the design was also a compromise between the number of monitor channels and the physical size of the device. (Robjohns, 1997: sec. 6, para. 1–3) Commonly, split consoles offered only 8-track or 16-track monitoring. In-line design approach did not have such a restriction: the monitor controls were included in every input channel strip. (White, 1994: sec. 6, para. 3)

Most of the affordable in-line consoles did have a specific restriction, however: the main channel shared a single equaliser and a single set of auxiliary sends with the monitor path. A splittable equaliser design provided a way to circumvent this restriction partially; the design offered a possibility to assign high and low filters to one of the signal paths and the mid filters to the other. (White, 1994: sec. 6, para. 4) This approach was used in, for example, Solid State Logic SL 4000 G (Solid State Logic, 1988).

Although analogue mixing consoles are still being produced, especially for live sound use, the large-scale studio consoles that once dominated the recording studios throughout the audio industry can be regarded as historical. The modern mixing consoles may also include features unavailable in the original analogue consoles. For example, Solid State Logic Duality SE (Solid State Logic, 2013a) and AWS (Solid State Logic, 2013b) consoles offer modern features, e.g. control surface functionality. Therefore, these products evidently target the modern production environments, and are in that sense very different from the traditional analogue-only consoles.

3 Examination of established user interface paradigms

Current computer-based digital audio workstation products are in many ways very similar to each other: there are prominent functional commonalities, such as timeline-oriented visual editing of audio and plug-in-based effect processing. In addition, certain interface metaphors, e.g. the analogue mixing console-based views, are ubiquitous. As discussed in the section 2.4, the history of audio production devices reveals that many of these common properties date conceptually back to the era before the advent of computer-based audio systems.

Established user interface paradigms in digital audio workstations are discussed in this Chapter. The focus is on the high-level interface conventions that characterise the current digital audio workstations. In other words, the main aim in this examination is to inspect the broad, fundamental interface structures, and to study the regularities defining the digital audio workstation interfaces.

The section 3.1 "Starting point" describes the initial observations that give support for this examination. As discussed in Chapter 2, sequencers, multitrack recorders, and analogue mixing consoles have provided the basis for the current digital audio workstations; the two fundamental user interface paradigms based on this background are discussed in the sections 3.2 "Track-based timeline view" and 3.3 "Mixing console view".

Metering – although a very different paradigm from the timeline and the mixing console views – is also an important subject in this examination, as metering is still one of the most fundamental visual monitoring methods offered in digital audio workstations. Therefore, concepts relating to metering are discussed separately in the section 3.4 "Metering". Lastly, as common personal computing is no longer solely based on the mouse and the keyboard, specific properties of touchscreen devices are discussed briefly in the section 3.5 "Attributes of touchscreen devices".

3.1 Starting point

The concepts are discussed in this examination at a high level of abstraction. This approach is selected, as it supports the main purpose of this examination – to offer a description of the fundamental interface paradigms ubiquitous in current digital audio workstations. The aim of this examination is therefore not to discuss individual interface elements extensively nor to inspect the graphic design.

Visual abstractions are presented to illustrate the discussed concepts. Abstraction is used instead of interface screenshots for several reasons. Firstly, abstractions allow portraying commonalities and dismissing the inessentials with simplified representations that can be interpreted swiftly. Secondly, formulating the abstractions requires reasoning and therefore reveals the central aspects in the paradigms – a major aim in this examination. Consequently, the approach contributes to the development of the modernised interface concept proposed in Chapter 4 "Processing with blocks – an interface concept for mixing". Lastly, few screenshots are reproduced in this thesis due to copyright reasons[5].

The following subsections offer an introduction to the examination presented in this Chapter. The subsection 3.1.1 "Incentive for the examination" presents some anecdotal observations that provide encouragement for the inspection of the digital audio workstation interfaces. The subsection 3.1.2 "Delimitation" outlines briefly the reasons why the particular paradigms were selected for the examination. Lastly, the subsection 3.1.3 "Methodology" concludes the introduction to the examination with a description of the methodological approach used.

5 As regards the most demonstrative DAWs in terms of the interface concepts discussed in this Chapter, a recent version of Avid Pro Tools and recent trial versions of MOTU Digital Performer and Steinberg Cubase were available for the examination. Unfortunately, only Steinberg granted a permission to use software screenshots in this thesis.

3.1.1 Incentive for the examination

Certain user interaction paradigms have become ubiquitous during approximately 25 years of digital audio workstation development. Interestingly, many fundamental user interaction paradigms in modern DAWs have remained largely untouched since the early computer-based audio workstations. Although some novel features have been developed in recent years, these features tend to stay proprietary for certain software, or take a long time to become established and shared across different workstations.

Tuning audio material provides a clear demonstration of the slowness in adapting new approaches. Celemony released its timing and pitch correction tool Melodyne on to the market already in 2001, following with a plug-in version in 2007 (Celemony Software, 2013a); highly flexible and well-visualised note-per-note offline tuning has therefore been possible for over a decade.

Although similar functionality is today integrated into multiple digital audio workstations, this has happened gradually. For example, Melodyne-like tuning was introduced to Apple Logic Pro X very recently, in 2013 (Apple, 2013c: 20, 396–402), and Pro Tools – one of the most common studio production workstations – notably still omits such functionality. This is despite the fact that tuning of especially vocal material is considered a standard production method in many current music genres. Even the manipulation of single notes within polyphonic audio recordings has been possible with Melodyne for the past five years (Celemony Software, 2013b), which further emphasises the slow progress in DAWs.

The developmental attitude seems to have been to accept digital audio workstation as a tool working principally in a certain way; instead of changing the basic dynamics, successive versions introduce small refinements. Naturally, the DAW manufacturers have reasons to guard the differentiating features and avoid any patent disputes. It is reasonable to assume that such reasoning might prevent some of the new features from becoming widespread. In addition, successful new features are not trivial to develop.

A joint effect of several factors makes examining the interface paradigms in digital audio workstations topical. Firstly, as described above, there has not been much innovation in the development of the digital audio workstations lately. Secondly, everyday computing has gone through a tremendous change since the concept of digital audio workstation was developed. Lastly, although the shifts in the studio

workflow and the music industry are not focal aspects of this thesis, there is arguably a close relationship between them and the information technological changes that have happened recently. On one hand, the gap between consumer-grade tools and professional equipment has narrowed, and on the other hand, changes in the professional production styles and genres demand up-to-date tools. In general, the long-standing analogies and metaphors are ripe for re-evaluation.

3.1.2 Delimitation

The user interfaces in digital audio workstations are, in essence, two-sided. For example, Apple Logic Pro (Apple, 2008; 2013c), MOTU Digital Performer (MOTU, 2011; 2013), Avid Pro Tools (Avid Technology, 2011a; 2013c), and Steinberg Cubase (Steinberg Media Technologies, 2013b) all include two traditional and distinct views: the track-based timeline view[6] and the mixing console view.

Although current digital audio workstations usually include also multiple specialised views for specific tasks, the fundamental workflow is in many ways based on the basic views. In current DAWs, these views provide the means for many traditional sound production procedures, such as recording, cutting and splicing recorded material, adjusting the channel levels or "volumes", connecting virtual plug-in effect devices, etc. The timeline view combines the multitrack tape concept with the sequencer concept and forms one part of the main interface paradigm. The mixing console view, or simply mixer view, forms the other. The interface is therefore based on the traditional hardware-based studio environment comprising specialised devices.

The timeline view and the mixing console view are commonly the primary ones for working with audio[7], but not necessarily for work based on manipulating note data. However, the other views common in digital audio workstations – including the note data editors, such as the piano roll view and the score view – are not discussed in this thesis. A general outline of the digital audio workstation views is available in, for example, the text by Nash (2011).

The focus of this examination is on the two distinct, established paradigms: the track-based timeline view and the mixing console view. The inspection of the mixing console paradigm is emphasised, as it provides the foundation for the

6 Sometimes referred to as "arrange view", "edit view", or "project view".
7 "Audio" refers here to rendered audio files and related signal processing.

modernised mixing user interface proposed in Chapter 4 "Processing with blocks – an interface concept for mixing". Approaching the interface examination from a different angle would likely result in a different set of prominent paradigms; the factors constituting current digital audio workstations could be weighed differently to include, for instance, the paradigm of editing.

Additionally, a thorough interface paradigm examination requires taking the concept of metering into account. Listening the auditory monitoring is arguably of paramount importance during recording and mixing, but metering provides a frame of reference: proper metering is vital for interpreting the levels of different tracks or channels in relation to fixed boundaries, such as the limits of the hardware in use. Consequently, the user interface elements relating to metering typically take up substantial area in the mixing console views, and in addition, most timeline views include metering in some form. The discussion of metering is restricted in this thesis strictly to aspects related to the relationship between the current computer technology and the traditional visual meter representations.

3.1.3 Methodology

A major thread in this examination is to inspect the user interface paradigms from the standpoint of the present day. The purpose of this examination is to examine the optimality, usability, and implications of the established paradigms. However, the aim here is not to create an exhaustive list of usability problems; instead, the aim is to describe the characteristics of the current digital audio workstation interfaces on a general level.

The traditional concepts of usability discussed in the section 2.3 "Concepts of interaction and usability" provide a frame of reference for the examination, and the concepts of heuristics form the main standpoint. Nielsen and Molich (1990: 249) describe four distinct ways to evaluate user interfaces:

> "There are basically four ways to evaluate a user interface: *Formally* by some analysis technique, *automatically* by a computerized procedure, *empirically* by experiments with test users, and *heuristically* by simply looking at the interface and passing judgement according to ones [sic] own opinion." (italics in the original)

This examination is based on the principles of heuristic evaluation, but instead of conducting a traditional usability study, the aim is to inspect the paradigms and to distil them into descriptive attributes. Therefore, the notion of heuristic evaluation is used mainly as a tool to approach the interfaces. In other words, this examination is based on carrying out observations on the interfaces, and contextualising them.

The heuristic evaluation method is not completely trouble-free: Nielsen and Molich (1990: 249) found in their four experiments that individual evaluators were able to find only half of the usability problems even in the best case. However, this weakness relates mainly to usability evaluations, and as this examination is primarily a descriptive one, searching for the maximum amount of usability issues is not the aim here. Moreover, Jeffries, et al. (1991) found that compared to other techniques – namely software guidelines, cognitive walkthroughs, and usability testing – heuristic evaluation produced the best results at the lowest cost.

The primary aim in this examination is to gather information about the central interface paradigms and to present this information in a condensed, abstract form. This result is then discussed in Chapter 4 "Processing with blocks – an interface concept for mixing": the subsection 4.1.1 "Problems of the established interface paradigms" provides an overview of the examined paradigms, and the observations are used as the basis for the interface concept.

3.2 Track-based timeline view

A typical timeline view in a current digital audio workstation consists of stacked horizontal tracks which represent audio signals, musical data, or other control data in relation to time. The track area typically incorporates scrollable and zoomable content representations, and at least level and panning controls pinned on one side of the view.

Other common user interface elements in timeline views are – among others – transport controls, time counters, buttons for selecting editing tools, zoom factor controls, and toggle switches for modal features such as grid-based editing. Additionally, some digital audio workstations include an "inspector" element which displays properties of the selected track; in essence, this connects the track view with the mixing console paradigm. An abstraction of the track-based timeline paradigm is shown in Figure 5.

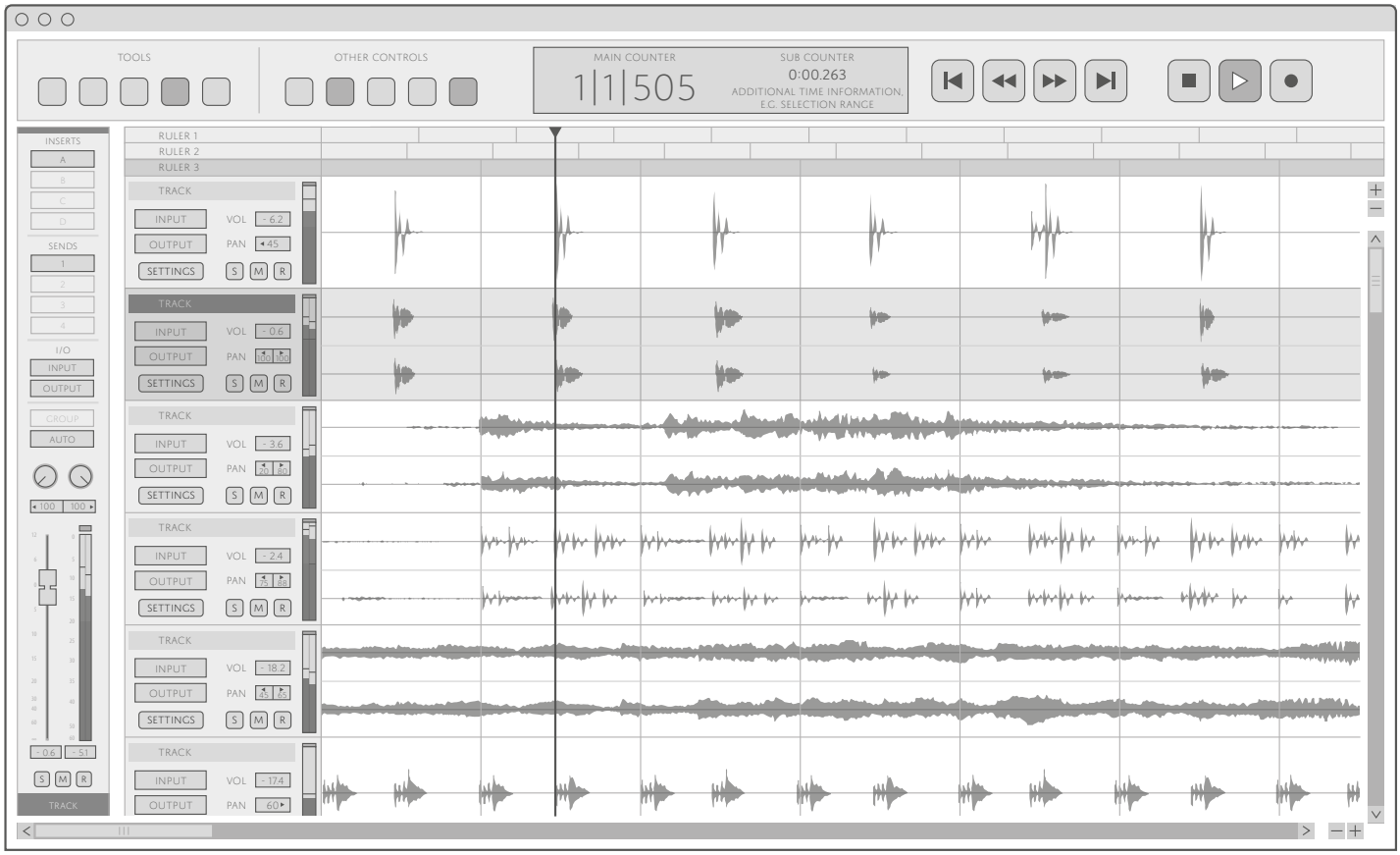


Figure 5: An abstraction of the established track-based timeline view.

The track-based division of audio material is reasonable from the standpoint of traditional Western music: tracks resemble staves. This creates a mapping between an instrument and a track – a concept analogous to traditional musical notation. Tracks can therefore easily represent instrumentations.

Essential properties of the track-based view are discussed in the consecutive subsections. The subsection 3.2.1 "Time in the track-based view" outlines the implications of the linear timeline. The signal representation – a dominating property of the track-based timeline views in digital audio workstations – is overviewed in the subsection 3.2.2 "Signal representation". Lastly, some typical restrictions of the timeline views are discussed in the subsection 3.2.3 "Visualisation of real-time processes".

3.2.1 Time in the track-based view

Time is represented in digital audio workstations in a linear manner. This is – despite the name – typical for non-linear editing (NLE) systems. The "non-linear" in NLEs commonly refers to the non-destructive re-organising of material. Morris (1999: 12) describes that the advantages of non-linear video editing systems include the flexible cut and paste functionality, the sophisticated tools for editing the material, the possibility to use multiple audio and video tracks, and the overall cost-effectiveness. Digital audio workstations largely share the same advantages.

Ironically, the biggest issues with typical NLE user interfaces become apparent when dealing with material that is essentially non-linear. A timeline is a very rigid boundary, which supports sequenced arranging, but discourages arrangements of other forms. Using random or probabilistic processes for determining the playback of specific sounds, or creating drastically different versions of the arrangement is usually tedious, restricted, or impossible in timeline-based interfaces.

The restrictions of the sequence-oriented approach are prominent in most major digital audio workstations. One of the most notable exceptions is Ableton Live, which breaks the common timeline–mixer model. In Live, the mixer is replaced with *Session View*, fundamental attribute of which is loop triggering (Ableton, 2013). This allows creating arrangements and compositions that are more freeform and non-linear than the ones the traditional timeline approach enables. Cakewalk Sonar now includes *Matrix View* which is meant for triggering loops and other sound events (Cakewalk, 2013b), and therefore offers functionality comparable to Live. Image-Line FL Studio is also less dependent on the timeline, due to its background. The initial versions of FL Studio were purely step sequencer-based (Image-Line Software, 2013a).

Still, when the user wants to save the trigger-based compositions, the typical way to achieve this is to record the performance into a traditional timeline. The paradigm within such software is therefore mostly performance-oriented and allows flexible experimentation, but does not change the fundamental way the digital audio workstation-based music production works.

3.2.2 Signal representation

Digital signal processing involves two essential representations of signals: the time representation, i.e. the waveform, and the frequency representation, in other words, the spectrum (Rossing, Moore, and Wheeler, 2002: 634). The discrete Fourier transform (DFT) offers a method to move between the domains in case of digital signals (Rossing, Moore, and Wheeler, 2002: 639). These concepts underlie the timeline views in digital audio workstations.

The field of digital signal processing is quite extensive, but discussing the subject is not especially relevant to the examination presented in this thesis. Moreover, the fundamental concepts of digital signal processing have been covered thoroughly in the literature; see, e.g. Oppenheim and Schafer (1975) and Smith (2007). The main interest here is to inspect the implications of the signal time domain representation, which allows displaying the signals over time as waveforms.

Typical time domain representation works well together with the track-based timeline: adjusting event timing in relation to the context is easy, as tracks are displayed as a stack. Keeping the phase relationship on multiple tracks unchanged is often crucial to avoid any unwanted phase-related issues when signals are summed. Fortunately, current digital audio workstations usually allow grouping selected tracks in order to cater for the editing needs of, for example, projects recorded with multiple microphones.

A remarkable facet of the timeline views in digital audio workstations is that the user is able to do both high-level editing and low-level editing within the same interface with the same methods. Zooming out allows the high-level structure of the project to be viewed and edited, while zooming in allows low-level editing down to the sample-level. Therefore, the timeline view is essentially a zooming interface. Moreover, zooming does not change the way the editing and selection tools work, which makes changing the level of zoom effortless.

To an experienced user, the time domain signal representation gives a reasonable idea of the materials' sonic properties. For example, silent parts are especially easy to find in this representation. Many common editing tasks, such as separating individual phrases from the recorded material, are therefore very quick to execute even without any auditory monitoring. However, the traditional waveform representation provides very little information about subtler qualitative attributes of the sound material, e.g. the timbre of the sound.

3.2.3 Visualisation of real-time processes

Waveform representation in digital audio workstations does a fair job in visualising the audio, but its usefulness is severely impaired by the lack of development. Tracks designed only for online[8] audio processing – auxiliary tracks, for example – do not visualise their commitment to the signal, as the processes happen in real-time. In fact, these tracks form a problem for efficient use of screen area: often, in a timeline view, auxiliary tracks display much less information than the regular audio tracks which contain the audio regions[9].

In some software, certain tracks can be excluded from the timeline view while keeping them visible in the mixer view. This approach, however, increases the need to jump between different views uncomfortably often. Moreover, displaying auxiliary tracks in the timeline view is useful for two main reasons: firstly, to keep the level controls and other essential parameters available at all times, and secondly, to display possible automation within these tracks. One might argue displaying the automation is reason enough for these tracks to take substantial space on the screen. Nevertheless, similarly to regular audio tracks, additional meaningful information could be visualised behind the automation curves.

Visualising auxiliary tracks beforehand exactly like rendered audio is evidently impossible without running intensive background processes. One solution to overcome this restriction would be to draw temporary time domain representations for online tracks during playback: this would still provide visual confirmation of the material just heard. Visualising the contribution of the un-rendered audio processes, such as processing happening in auxiliary tracks or within plug-in devices, would give complementary visual feedback useful in case of unreliable auditory feedback – audio monitoring systems, including the listening space acoustics, quite commonly lack in certain aspects of high fidelity playback.

A visualisation approach similar to the idea presented above has been used in Steinberg Nuendo digital audio workstation, although in the mixing console view instead of the waveform view. Displaying vertically scrolling waveforms has been a feature of the Nuendo's mixer view since the software version 5, albeit involving

8 The term "online" is used here in reference to processes that occur real-time, whereas "offline" refers to rendered files, e.g. recorded audio files.

9 "Region" is one of the commonly used terms for referring to a certain portion of a concrete audio file, not to the actual file itself.

only rendered audio (Steinberg Media Technologies, 2013c). The new Avid S6 control surface offers similar functionality (Avid Technology, 2013d).

Novel ideas for updating the waveform view have also been proposed in recent research. Gohlke, et al. (2010) presented prototype designs for a more efficient waveform view in their study: the prototypes demonstrated the use of overlaid images to represent the contributions of effect devices and, in addition, described a colour-coded, threshold-based view for illustrating the spectral positions of dominating tracks.

3.3 Mixing console view

The analogue mixing console is the basis for the de facto mixing environment in digital audio workstations: every prominent DAW includes a view based on this paradigm. The actual implementations vary slightly, but the common element in the current designs is the channel strip: the mixers consist of vertical strips which group together processing and routing options. These groups form the channel entities. An abstraction of this paradigm is shown in Figure 6.

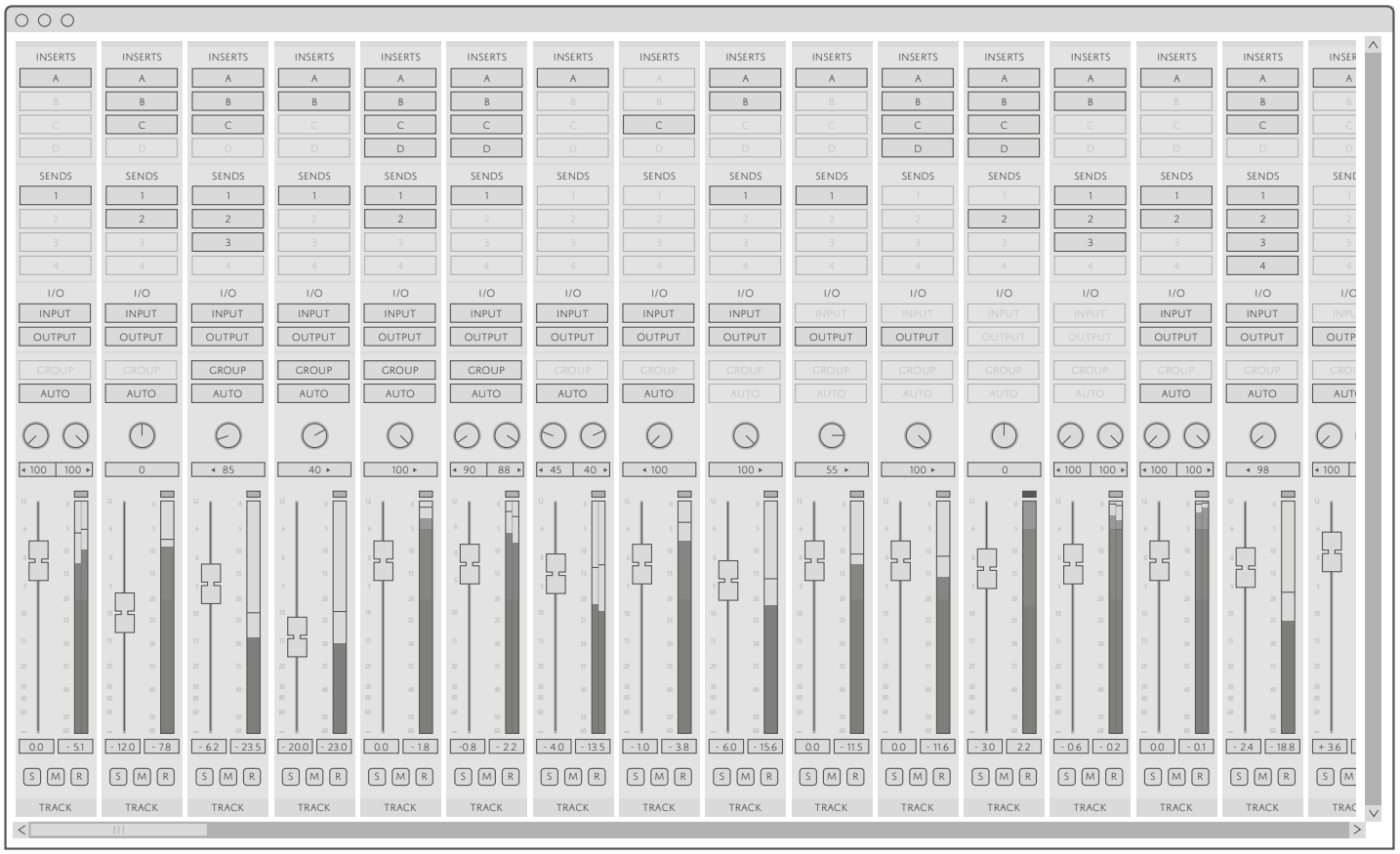


Figure 6: An abstraction of the established analogue mixing console-based view consisting of channel strips.

The subsection 3.3.1 "Channel concept" gives an overview of the fundamental ways channels are approached in current digital audio workstations. The parts that constitute the channels and the relevant routing methods are discussed in the subsection 3.3.2 "Channel elements and signal chain". The subsection 3.3.3 "Primary signal path" presents briefly a few key implications of the common channel structure.

3.3.1 Channel concept

The concepts of track and channel are central to audio workstations. The track–channel relationship is approached essentially in two different ways in digital audio workstations:

1. Tracks and channels are different representations of the same concept;

2. Tracks are sources for channels.

Avid Pro Tools is an example of the former approach: many channel properties are accessible in Pro Tools from both *Edit Window* (i.e. the timeline view) and *Mix Window* (i.e. the mixing console view). In fact, the term "track" is used in Pro Tools in reference to this single entity comprising both the traditional timeline 'track' and the channel strip. (Avid Technology, 2013c)

Apple Logic Pro uses the latter approach. Terminologically, the track entities in *Logic Pro main window*[10] (i.e. the timeline view) are called *tracks*, whereas channels in *Mixer* (i.e. the mixing console view) are referred to as *channel strips* (Apple, 2013c: 22, 25). By default, creating new tracks in Logic Pro also creates corresponding channel strips automatically (Apple, 2013c: 163). However, multiple tracks can use the same channel strip (Apple, 2013c: 166).

These approaches offer mostly the same functionality, as most DAWs have a hidden audio routing structure, which allows routing signals inside the system from a channel to another through "buses"[11]. Buses can be used as "virtual" inputs and outputs, which allows flexible signal routing inside the DAW. This routing system

10 Apple renamed the Logic Pro timeline view in Logic Pro X, i.e. the newest version of the software. The term Logic Pro main window is used in Logic Pro X, while the term Arrange window was used in the prior versions. (Apple, 2013c: 67)

11 The spelling "bus" is used in this thesis. In some literature, the same concept is also referred to as "buss".

is largely similar to the one used in the analogue mixing consoles; however, the digital environment allows much greater number of buses.

Interestingly, recent developments in digital audio workstation mixing views have brought the mixing environments even closer to the analogue mixing console paradigm. Cubase 7 – a recent version of the prominent digital audio workstation – introduced *MixConsole*, an even more literal recreation of the analogue mixing console paradigm. In *MixConsole*, specific equalisers, dynamics processors, and saturation effects are built directly into the channel strips, which offers direct access to the essential effect parameters (Steinberg Media Technologies, 2013b). A *MixConsole* channel is shown in Figure 7 (Steinberg Media Technologies, 2013b). Cakewalk Sonar offers similar functionality with its feature *ProChannel*, albeit with a different approach. Sonar's *ProChannel* is an additional strip incorporating specific proprietary effects (Cakewalk, 2013c).

Basing the audio software on recreations of analogue devices is not a new idea, however; Propellerhead Reason is possibly the most famous example of such analogue environment-modelling software. Reason is based on instrument racks and virtual patching cables, and many of its user interface elements are quasi-replicas of their hardware counterparts (Propellerhead Software, 2013). Software audio processing plug-ins offer also digital reproductions of analogue devices; some of these plug-ins even incorporate the interference characteristics present in the analogue devices, e.g. electric hum (Waves, 2013a; Waves, 2013b).

3.3.2 Channel elements and signal chain

A typical digital audio workstation channel is divided into specific audio processing and routing elements: the input, the effects, the sends, the level controls, the panning controls, and the output – a structure quite similar to the analogue mixing console channel structure. The channel meter offers visual feedback by providing information on the signal level. In addition, channels usually offer controls for at least the channel automation and the group assignment, and specific track types may include additional elements, such as MIDI effects and virtual instrument controls.

Figure 7: A channel strip in Cubase 7 MixConsole (Steinberg Media Technologies, 2013b). Screen captured by Petri Myllys. Used with permission of Steinberg Media Technologies.

The signal flow[12] describes how a signal travels in a system. The signal flow in digital audio workstations is in many aspects very similar to the signal flow in the analogue mixing consoles: individual input signals travel through channels that include specific routing and processing elements. The user has some control over the order of these elements, but commonly this control is restricted. After passing through the channels, the individual signals are summed together. Multiple such summing stages are often used for managing a large number of channels.

This subsection provides an overview of the essential channel elements typically included in the digital audio workstation mixing console views. The channel meters – although integral to the channels – are not discussed here; instead, metering is inspected separately in the section 3.4 "Metering". A schematisation of a conventional DAW channel, illustrating the signal flow, is shown in Figure 8.

Input

To initiate the signal flow in a channel, a signal needs to be inputted. This input signal can be either a monitored external source, e.g. an acoustic sound source captured with a microphone, or an internal source – namely a rendered audio file or a signal routed from another track. Channels in digital audio workstations commonly incorporate drop-down lists for the input selection, and buttons for controlling whether the selected input or the associated track is monitored.

Effects

The inputted signal is first processed with the channel effects, if applicable. Effect processing in digital audio workstations is characterised by encapsulation: the effects are usually placed in so-called "insert slots" in DAWs – a term derived from the analogue way of using outboard effect devices by inserting them into the signal path. Whereas in the analogue environment physical restrictions apply (e.g. insert sockets are limited in number as are outboard devices), the digital audio environment could be virtually free of such limitations. Yet, some digital audio workstations offer only a limited number of these slots (Avid Technology, 2013c; Steinberg Media Technologies, 2013b), which negates one of the flexible aspects of the digital audio environment.

12 The signal flow is inspected here from the user's standpoint. Therefore, the described flow does not necessarily reflect the technical implementation of the audio processing.

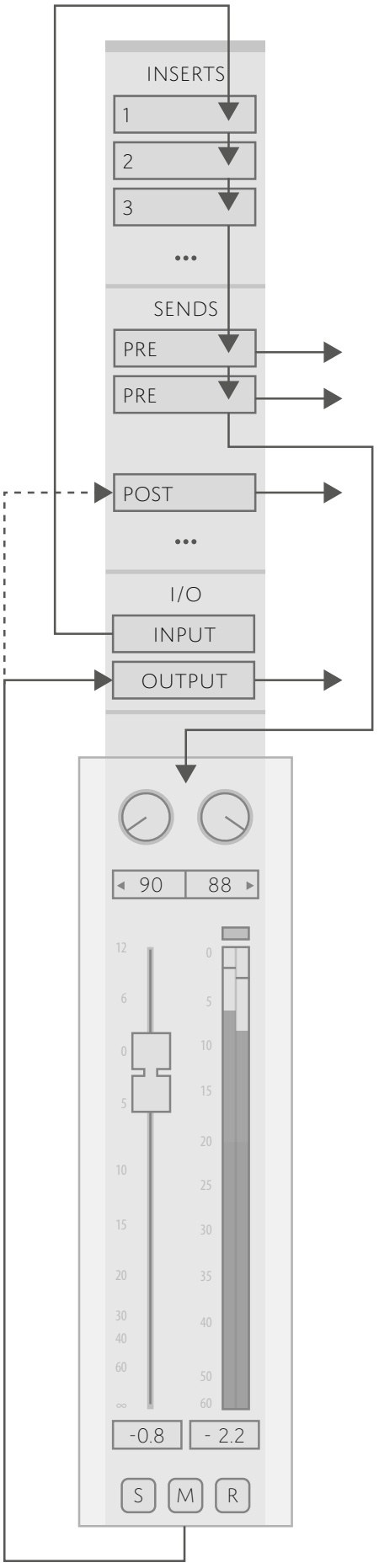


Figure 8: A schematisation of a conventional, console-based digital audio workstation channel.

Miniature equaliser curves can be displayed in some digital audio workstations, for example in Apple Logic Pro (Apple, 2008), and in Steinberg Cubase (Steinberg Media Technologies, 2013b). However, other effects are commonly contained in generic effect boxes displaying only the effect names, although improvements have been made recently. For example, Avid Pro Tools HD 11 can display gain reduction meters inside the *Insert Assignment* boxes of specific dynamics plug-ins (Avid Technology, 2013e). Steinberg Cubase and Cakewalk Sonar, on the other hand, allow controlling certain effect parameters without opening plug-in windows (see the subsection 3.3.1 "Channel concept").

Ableton Live, introduced in 2001 (Henke, 201?), offers a distinctly different approach. Live includes a variety of *devices* that can be used for the creation of device chains (Ableton, 2013). The chains can be inspected in the *Device View* which supports expanding and collapsing individual devices on the spot (Ableton, 2013). This allows inspecting the signal flow and processing in a way which is fundamentally different from the typical encapsulated effect slot principle. A device chain consisting of *EQ Eight*, *Compressor*, *Saturator*, and *Simple Delay* devices is shown within the Ableton Live interface in Figure 9 (Ableton, 2013).

Figure 9: Ableton Live interface with "Device View" (the bottom pane) opened (Ableton, 2013). Screen captured by Petri Myllys. Used with permission of Ableton.

In addition to the device chains, Live offers another significant feature as concerns effect handling – and plug-ins in general: certain devices can contain other devices. *Effect Racks* and *Instrument Racks* are, in essence, container devices that offer a way to create multiple parallel chains within a track (Ableton, 2013). Therefore, Live allows parallel processing within individual channels – a procedure not possible in the typical mixing console-based interfaces. Live also includes *Drum Racks* and *MIDI Effect Racks* (Ableton, 2013).

Sends

After the effects, the processed signal can be sent to other channels. In some digital audio workstations – e.g. in Pro Tools – the signal can be sent to physical outputs or "virtual" buses (Avid Technology, 2013c). Buses can be selected as inputs for auxiliary channels, which results in a send–return-relationship similar to the auxiliaries in the analogue mixing consoles. In some other DAWs, e.g. in Ableton Live, such routings are done without the bus structure by sending the signal directly to a return channel (Ableton, 2013). Abstractions of these approaches are shown in Figures 10 and 11.

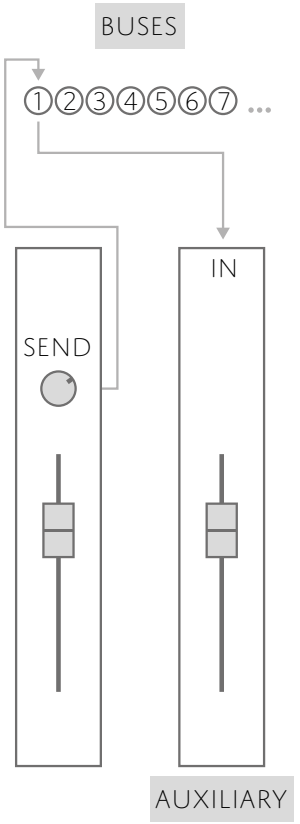


Figure 10: An abstraction of a parallel processing setup in a digital audio workstation employing a bus structure.

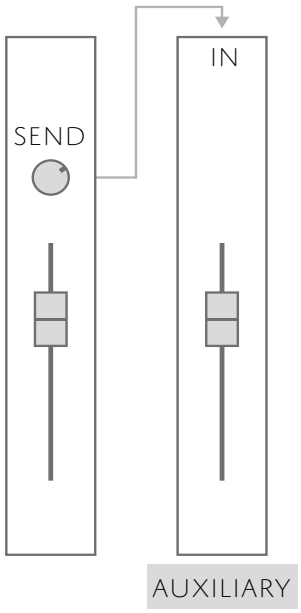


Figure 11: A parallel processing setup in a digital audio workstation employing a direct routing system.

"Send effects" and other forms of parallel processing are not the only use for the sends; in addition, signal sends are often used also for creating separate monitor mixes for the artists. Controlling the signal level sent on the spot is crucial for this task to be manageable, and it is thus not surprising that such functionality is included in practically every major digital audio workstation. However, Avid Pro Tools offers a curious case. Only the most recent version of Pro Tools allows controlling more than two sends per channel simultaneously without opening separate windows (Avid Technology, 2013f).

In addition to parallel processing and monitor mixing, there is a third distinct use for the sends: routing signals to effect side-chain inputs. This reveals the fundamental differences between the bus-based routing structure and the direct routing approach. Setting up a side-chain input in Ableton Live does not require sending the signal at all, as channel outputs can be directly selected as side-chain inputs (Ableton, 2013). This approach has an evident advantage over the bus-structure: achieving a similar result requires one less step. However, the approach can also quickly result in an unwieldy list of available side-chain inputs, as every channel output is automatically included as an option.

Panning and level controls

Before reaching its destination, e.g. the main output or a sum channel input, the signal goes through two crucial channel elements: a gain stage and a panning control. The gain stage determines the "channel volume" – controlled with a level fader, and the panning controls are used to place the signal at a specific "location"[13] in a multichannel setup.

The level faders in digital audio workstations are sliders that usually move vertically, a concept transferred very literally from the analogue mixing consoles. It is no wonder faders have remained ubiquitous even in software: in terms of affordance (see the section 2.3 "Concepts of interaction and usability"), it is easy for the user to grasp how faders are operated.

The panning controls have three common interface element types: (1) the panning knob or the "pan pot" which is equivalent to the analogue potentiometer, (2) the horizontal panning slider, and (3) the two-dimensional "surround panner".

13 The actual effect the panning controls have on the signal depends on many factors, including the channel setup and the panning laws used.

Whereas a physical knob evidently affords rotational movement, the mapping related to the mouse complicates the interaction significantly. Therefore, in comparison to faders, the knob-style graphical user interface elements offer poor affordance.

Examining the panning controls more closely reveals that there are two distinct implementations for the stereo panning controls: a single control which is essentially a stereo "balance" control, and dual controls which allow controlling the panning of both channels individually. The dual control type offers superior flexibility, as it allows narrowing the width of a stereo source, and panning the result. Yet, the balance type is still the only choice in, for example, the mixer of the newest version of Apple Logic Pro[14] (Apple, 2013c); naturally, the single-knob approach does have the advantage of being simpler.

Output

The signal needs to eventually "flow" out of the channel. The interface elements for the outputs are in digital audio workstation mixing console views typically similar to the inputs: drop-down lists of available physical and virtual outputs. A standard way to control the mix especially in large sessions is to output individual audio channels into sum channels, which are then outputted into further sum channels or the main outputs. The bus structure allows such routings, and in some digital audio workstations channels can be grouped into folders that may do such routings automatically.

3.3.3 Primary signal path

The common visual channel strip layout contradicts the actual signal flow. The effect slots suggest a vertical, descending signal flow representation: the signal flows in most DAWs through the stack of effects one by one starting from the topmost effect, and the signal is passed to other channel elements only after all the effect processing is done[15]. However, some channel strip elements break this conception.

14 A plug-in offering more flexible panning is included in Logic Pro, which provides an alternative to the balance-type panning (Apple, 2013c).

15 There are also exceptions to this flow: in Steinberg Cubase, the effects in Insert effect slots 7 and 8 are post-EQ and post-fader (Steinberg Media Technologies, 2013b).

Firstly, the input selector is not necessarily visually the first channel element, i.e. the topmost one, nor is the output selector the last. For example, in Avid Pro Tools, the input and output selectors are grouped together to form an "I/O" element, which is positioned below the effects and the sends, and above the level controls and the panning controls (Avid Technology, 2013c). Secondly, the position of a send is often visually misrepresented in relation to the actual signal path; if a send is set to a post-fader position, its typical visual location above the channel fader no longer corresponds the position on the signal path.

Interestingly, the user has so little control of the signal flow happening inside a channel that although the visual element order does not correspond the actual order of the flow, new users likely learn the signal flow without any severe problems. The control the user has over the order of the channel elements is usually restricted to the following aspects: changing the order of the effects and setting the individual signal sends to either pre-fader or post-fader positions.

According to Nielsen (1993: 26), five attributes are traditionally associated with usability: learnability, efficiency, memorability, errors, and satisfaction. The most severe restrictions of the mixing console view do not seem to relate to learnability, but instead, they concern efficiency. Software does not have the physical restrictions of the analogue mixing consoles that require the mixer sections and controls to be situated in certain, fixed places. It would be quite possible to design – instead of replicating the restrictive analogue structure – a digital audio workstation mixer that would allow the user to make use of the wide array of signal paths available in a digital system.

3.4 Metering

Metering provides arguably the most important visual feedback of signal levels in recording and mixing systems. In analogue domain, metering is the main non-auditory cue available (in addition to the means related to the signal indirectly, e.g. time codes and observing the fader positions). An oscilloscope may be used to complement metering in analogue systems, but as this is commonly not a built-in device in recording devices or mixing consoles, signal level meters ought to provide enough information for typical recording and mixing situations.

Complementary to listening in many ways, challenges relating to metering are very different from those of auditory monitoring. While the optimal audio

reproduction can be thought as a faithful playback of every signal component, this approach is not valid for visual meters. "Meter", in general, refers to "a device that measures and records the quantity, degree, or rate of something" (Oxford University Press, 2005). An obvious quantity to represent in an audio meter is the signal amplitude, or more specifically, a certain measurement or an approximation of the amplitude. This poses a substantial problem for the meter design: a meter relying on a single representation will inherently only show some particularities of the signal.

The general graphical user interface approach to metering in modern digital audio workstations is presented in the subsection 3.4.1 "Interface elements". The relationship between the elements and the capabilities of modern audio processing systems is discussed in the subsection 3.4.2 "Implications of the digital domain". The function of this overview is therefore not to provide an in-depth inspection of metering; instead, the aim is to present the problematics of traditional metering in current, advanced digital audio workstations. The focus is on the measurements related to the limits of the system, namely the full scale metering. Aspects such as psychoacoustics, signal measurement time windows, meter ballistics, loudness measurements, and display performance are intentionally omitted from the discussion.

3.4.1 Interface elements

Modern digital audio workstations typically employ, in essence, very traditional presentation for the meters. A meter in the mixing console view is commonly a vertical bar marked with values and having an interior colour which extends along the bar as the signal level increases. Displaying meters in the timeline view in addition to the mixing console view is common, and there are essentially two ways in which the meters are displayed in the limited space the track-based views offer. The meters may be displayed vertically adjacent to the controls of the associated tracks (Avid Technology, 2013c), or the meters may be rotated 90° to situate them between the tracks (Cockos, 2013).

Metering has commonly two essential modes[16]: pre-fader metering and post-fader metering. The terms are – due to the analogue mixing console-like design of the interface – rather self-explanatory: channels incorporate fader elements for

16 Different scales result in a number of additional modes, but these differ from the modes presented here, and are not part of this examination.

level adjustment, and the mode allows selecting the level measurement point on the signal path. This modality is reasonable, as both modes are effective in certain situations. For example, pre-fader metering is practical when recording to a digital audio workstation and controlling the monitor mix with the channel faders, as pre-fader metering prevents the mixing adjustments from affecting the displayed input levels. Post-fader metering is similarly useful when visual feedback from the level adjustments is desirable, e.g. during many mixing tasks.

The meters are often divided into colour-coded areas (Avid Technology, 2011a; Apple, 2013c; Avid Technology, 2013c). Segmenting the meter with different colours provides at-a-glance information. The segments may be user-definable as they are in, for example, Steinberg Cubase (Steinberg Media Technologies, 2013b) and the newest Avid Pro Tools (Avid Technology, 2013f). In some software, e.g. in older versions of Pro Tools, the segments are fixed (Avid Technology, 2011a).

3.4.2 Implications of the digital domain

Volume unit (VU) metering is the traditional way to display signal levels in analogue equipment. The 0 VU refers to the standard operating level for most mixing consoles and tape recorders, and due to the VU meters' inability to display short-term peaks, professional consoles commonly provided decent *headroom* above the zero-point. (Huber and Runstein, 2005: 457) Overdriving analogue devices and tape machines slightly, and therefore gradually distorting the signal, was a technique used for changing the characteristics of the sound (Huber and Runstein, 2005: 424, 457–458; Robjohns, 2000: sec. 3, para. 1).

Digital systems do not share the principle of graceful saturation analogue recording offers. Instead, digital systems hard-clip[17] the signal when its amplitude is out of range, leading to a complete loss of information within the clipped region. This upper boundary of the dynamic range is referred to as zero decibel full scale, abbreviated to 0dBFS (Robjohns, 2000: sec. 4, para. 1).

Decibel (dB) itself is a dimensionless unit used for comparing two quantities of which one can be a fixed reference (Rossing, Moore, and Wheeler, 2002: 99, 118). In the case of dBFS, the fixed reference is the maximum representable level (Robjohns, 2000: sec. 4, para. 1). Internally, floating-point arithmetic is used

17 In some systems, this can be compensated to some extent by emulating different clipping behaviours.

for the audio processing in modern digital audio workstations (Avid Technology, 2011b; Cakewalk, 2013d). Discussing the consequences of such calculations goes beyond the scope of this thesis, but as a general result, the internal headroom in modern digital audio workstations is substantially higher than any music-related scenario requires.

The 0dBFS is therefore determined not by the digital audio workstation software, but by the most constrained link of the system. Commonly, this is the resolution of the file or the configuration or the performance of the analogue-to-digital or digital-to-analogue converter. The implication is that there are at least three potential scenarios for clipping to occur:

1. Recording – signal levels overloading the analogue-to-digital converter or the recording format;

2. Playing back – the accumulated level of summed signals overloading the digital-to-analogue converter;

3. Exporting[18] – insufficient destination file resolution for representing the exported signal.

Fortunately, audio devices available today typically offer adequately high dynamic ranges: the noise floors originating from these devices are often not the prime concerns. Therefore, clipping happening during recording is not likely if appropriate recording practices, such as reserving sufficient headroom, are exercised.

Clipping the signal during the playback is easily avoided by lowering the output level, but there is a substantial mismatch between the typical user interface meter element and the actual audio processing. As discussed above, modern digital audio workstations commonly offer great internal resolutions. However, the clip indicators on the individual audio channels – typically grouped together with the meters – commonly indicate clipping even when the summed level is set so that no clipping actually occurs.

Exporting is similarly affected. There have been recent improvements to this conventional behaviour, however. Avid Pro Tools 11 offers clip indicators that display different colours depending on whether clipping actually occurs, or if there is

18 Exporting is often also called "bouncing". The term refers here to the procedure of rendering audio files.

merely a possibility of clipping happening at a later stage of the signal chain (Avid Technology, 2013f: 30, 35).

The scenarios described are not the only situations in which signals may be exposed to clipping in digital audio workstations, but they demonstrate the juxtaposition of the full scale metering – which has remained largely unchanged – and the capabilities of the digital processing systems – which have evolved significantly. Although adjusting the individual signal levels in a way that no clipping – whether potential or actual – is indicated is often considered a good gain staging practice, an interface that misleads the user in this way can hardly be considered appropriate.

3.5 Attributes of touchscreen devices

Digital audio workstations have traditionally been associated with the long-standing desktop computing operating systems, namely Apple OS X and Microsoft Windows. However, as discussed in the section 2.2 "Brief review of the technological basis", compact touchscreen devices are becoming increasingly popular. This hints that a shift in mainstream personal computing is taking place.

The dichotomy between mobile devices and desktop devices is arguably behind the times, or more specifically, the division is not sufficient. Keyboards, touchscreens, and portability may be arranged in any number of combinations, resulting in distinct form-factors offering different interaction methods. In addition, the boundaries of operating systems are shifting. Prominent mobile operating systems offer features such as capable multitasking (Apple, 2013d; Google, 2013b) – traditionally associated with desktop computing. What is more, Microsoft Windows 8 supports, by design, both traditional input devices and touchscreens (Microsoft, 2013e).

The essential characteristics of touchscreen devices are outlined briefly in the subsection 3.5.1 "Implications of touchscreen-based interaction" from the standpoint of user interaction. Additionally, some sound production-specific aspects are remarked. The subsection 3.5.2 "Mobile digital audio workstations" provides an overview of the general qualities of current mobile digital audio workstations.

3.5.1 Implications of touchscreen-based interaction

A touchscreen combines the input device with the display device, and essentially removes one stage from the chain of human–machine interaction: whereas the movement of a mouse needs to be translated into relative pointer movement, a touchscreen allows touching – or *tapping* – the actual point of interest directly. Therefore, removing the mapping process inherent for mice has one clear advantage: the user is able to manipulate the system more directly.

Today, touchscreens are common in devices small and large. Mobile devices are compact enough to be held in the hand, which – in addition to the touch-based interaction – sets them apart from traditional desktop computing devices. Whereas using a computer has traditionally meant sitting in front of a desk in a more or less static posture, such restrictions do not apply to modern mobile devices. Laptop computers laid the ground for portable computing, but mobile phones and tablet computers introduced the concept of truly mobile computing.

Modern touchscreen devices are generally able to recognise and follow multiple individual touch points – a feature commonly referred to as multitouch. This enables controlling the device with a variety of gestures. Mainstream touchscreen devices capable of following multiple points of touch are still relatively new (Vogelstein, 2013: para. 31), but certain gestures have already become ubiquitous, e.g. two-fingered pinch-to-zoom gestures.

Large, modern touchscreens provide possibilities for updating desktop computing. Some audio-related applications are already available. For example, Slate Audio Raven consoles offer multitouch control for major digital audio workstations (Slate Pro Audio, 2013a; Slate Pro Audio, 2013b), while Cakewalk introduced support for multitouch in an update for Sonar X2 (Cakewalk, 2013e), demonstrating one way in which a digital audio workstation software can be combined with a regular touchscreen. Although virtually any modern digital audio workstation can be used with a touchscreen, the multitouch support is one of the key features the developers need to implement in order to offer capabilities products such as Raven and Sonar offer.

There are factors that – if emphasised – put the touchscreen-based interaction at a disadvantage. One of these factors is especially prominent: the touch interaction has a limited accuracy. Whereas a typical mouse is an object that can be gripped and controlled with a great amount of precision, touching a glass surface with

fingertips does not allow such fine control. What is more, user interface elements need to be large enough to accommodate the inaccuracy of the touch event itself, i.e. the relatively large size of the fingertip (Parhi, Karlson, and Bederson, 2006; Park, et al., 2008).

Many graphical user interfaces are based on representations of physical objects: buttons, switches, knobs, sliders, etc. Still, users manipulate these objects through a medium which distorts the interaction. Current feedback methods for touchscreen devices – most commonly sound and vibration-based active haptic feedback – are incapable of accurately supporting the tangible qualities of the displayed objects. A tangible touchscreen – one of the holy grails of interaction design – could offer significantly better feedback. Concepts for such interfaces have in fact been presented in recent studies (Maschmeyer and Cameron, 2012; Kim, Israr, and Poupyrev, 2013). These studies indicate that the current touchscreen designs may be superseded in the future.

3.5.2 Mobile digital audio workstations

Recently, multiple music production applications have been released for various mobile platforms. These apps include performance-oriented tools, e.g. iMS20 synthesiser for Apple iPad (Korg, 2013) and KORG M01D music workstation for Nintendo 3DS (Detune, 2013). In addition, there are apps suitable for multifaceted music production, e.g. iPad recording and sequencing app Cubasis (Steinberg Media Technologies, 2013d) and FL Studio Mobile HD for Apple iOS and Google Android (Image-Line Software, 2013b).

Mobile digital audio workstation apps resemble traditional desktop digital audio workstations in many ways. For example, track-based timeline editing views are common also in the mobile DAWs (Blip Interactive, 2010; Apple, 2013e; Image-Line Software, 2013b). In addition to the similarities in the basic paradigm, there are also similar features.

Steinberg Cubasis offers mixing and editing environments with, for example, automation, insert effects, send effects, and support for routing signals between different apps (Steinberg Media Technologies, 2013d). WaveMachine Labs Auria includes support for third-party plug-ins ported from the traditional desktop environments (WaveMachine Labs, 2013). External audio interfaces and MIDI controllers are supported in many mobile audio workstations. In fact, the feature

sets available today are so extensive that comparing mobile DAWs with desktop DAWs results in a long list of similarities.

There are, however, also substantial differences between mobile digital audio workstations and traditional desktop DAWs. First, a different input device is used for operating the system. As discussed in the previous subsection, mobile devices offer intimate control, yet the accuracy of touch is inherently limited. The information density in the mobile digital audio workstations is therefore often lower than in the desktop DAWs. Sensing multiple points of touch simultaneously is also a key differentiator: whereas mouse-based operation is limited to adjusting a single control at a time (or resorting to modality), especially the tablet-sized touchscreen devices make controlling multiple channels simultaneously effortless.

It is possible to take advantage of the intimate multitouch-based interaction and yet offer the full capabilities of the traditional desktop digital audio workstation by separating the two aspects from each other. In other words, the touchscreen device can be used as a controller for the desktop application. Apple (2014) and MOTU (2014), for example, offer such companion apps for their respective desktop digital audio workstations. In addition, there are controller apps such as hexler TouchOSC (Fischer, 2013) that are more universal.

Whether mobile digital audio workstations will merge with the traditional, desktop digital audio workstations in the future remains to be seen. During the era of analogue studio environments, the specialised audio devices offered hardware tailor-made for audio production. Today, the computer-based audio production is dependent on the surrounding technological framework. Therefore, as long as this dependence is the foundation for the digital audio workstations, the developments in computer technology need to be monitored carefully.

4 Processing with blocks – an interface concept for mixing

The versatility of a properly designed analogue console was remarkable – the same hardware device was able to cater for very different situations. On the other hand, the general limitations of the physical devices and the strong industrial structure around the productions gave the sound engineering a format, which allowed the studio workflow to become established. The technological changes that happened during the era of the analogue studio workflow were not comparable to the recent advancements in computer technology, and the status quo was maintained.

The notion of audio production, created by the analogue studio workflow, still existed during the development of the early digital audio workstations. The core of this concept is to propose a modernised approach to mixing in digital audio workstations. Therefore, the question whether the established interface paradigms are still optimal – the topic central to this thesis – is discussed in this Chapter.

The premise of the interface concept is discussed in the section 4.1 "Starting point for the concept". The main concept presentation is divided into three subjects. Firstly, the structural aspects of the interface are outlined in the section 4.2 "Interface structure". Secondly, the features central to the interface concept are

discussed in the section 4.3 “Principal features”, and thirdly, the interaction between the user and the interface is described in the section 4.4 “Interaction” with regard to the fundamental features of the concept.

In addition, as personal computer technology is constantly progressing, the scalability of the interface is discussed. Two distinct aspects of scalability are considered: supporting different input devices and providing for display devices of varying size and quality. These aspects and the related design methods are presented in the section 4.5 “Addressing different devices”.

4.1 Starting point for the concept

The common user interface approach in digital audio workstations has remained remarkably static. Computers brought a novel, visual way to manipulate note data and audio, but since this major step, substantial innovations have been thin on the ground. The other fundamental interface paradigm, i.e. the ubiquitous analogue mixing console-based view, demonstrates that the concept of mixing was established well before the advent of the DAW.

The audio processing capabilities of the modern digital audio workstations are immense. The severe restrictions of the early digital systems, e.g. limited headroom and insufficient processing power, are now either non-existent or substantially less significant than they used to be. Circumstances have changed, and the issues have shifted to different areas. The first digital audio workstations introduced significant new possibilities with the graphical user interface, but the lack of processing power limited this potential. Today, however, the situation is reverse: the current workstations offer a vast amount of processing power, but the inflexible user interface restricts the use of this power to the execution of conventional procedures.

The friction between the current technological capabilities and the interfacing conventions provides fertile ground for exploration. The strongly analogous interface structure common in current digital audio workstations is not necessarily optimal: for example, some common tasks require unnecessary steps, and complex signal routings may lead to configurations that are difficult to comprehend at a glance. What is more, these problems are cumulative: they reoccur often and complicate the mixing situation increasingly. This state of affairs provides the starting point and the incentive for this concept.

The subsection 4.1.1 "Problems of the established interface paradigms" provides a filtration of the prominent, restrictive aspects of the common interface structure inspected in Chapter 3. Specific objectives for the interface – based on the observations about the established paradigms – are described in the subsection 4.1.2 "Main aims of the interface concept". The general idea of the concept is presented in the subsection 4.1.3 "Proposal"; this gives an outline of the core ideas that are essential to the concept. The subsection 4.1.4 "Presentation" describes the way in which the interface concept is presented in this text.

4.1.1 Problems of the established interface paradigms

As discussed in Chapter 3, two major structural interface paradigms are prominent in the conventional digital audio workstation user interfaces: the track-based timeline view and the mixing console view. The track paradigm and the mixing console paradigm are the fundamental concepts that underlie many common sound production procedures. In addition, metering essentially forms a third established interface paradigm, and is inseparable from the sound engineering workflow.

New production styles have become apparent recently, likely due to the emergence of new music genres and changes in the structure of the industry. For example, to accomplish fresh results, modern popular music production may require methods that differ from the traditional procedures – there are modern tools and practices for concepts such as dynamic range compression, tuning, and editing. While the aesthetics of these new directions remain debatable, they do provide evidence that a shift in music production is occurring.

Thorough inspections of the conventional sound engineering practices, the current production methods, and the characteristics of current users would be necessary to draw any definite conclusions on the large-scale ramifications of the established digital audio workstation interface paradigms. Furthermore, the interplay between the traditional practices and the numerous essential recording and sound production devices – e.g. microphones, microphone preamplifiers, analogue-to-digital converters, and digital audio workstations – is so intimate that inspecting any one of these separately does not reveal the reasoning behind the established procedures. Nevertheless, a modern audio workstation should preferably not only support the traditional production methods, but also offer means to get new types of results easily.

The relation between the established user interface paradigms in digital audio workstations and state-of-the-art personal computers is another important aspect when assessing the status of the paradigms. This relationship is essentially the core of this thesis, and as discussed in the previous chapters, it provides support for the notion that current digital audio workstation designs have some inherent difficulties in addressing current needs, owing to the extensive use of analogies and interface metaphors. This situation is illustrated in Figure 12.

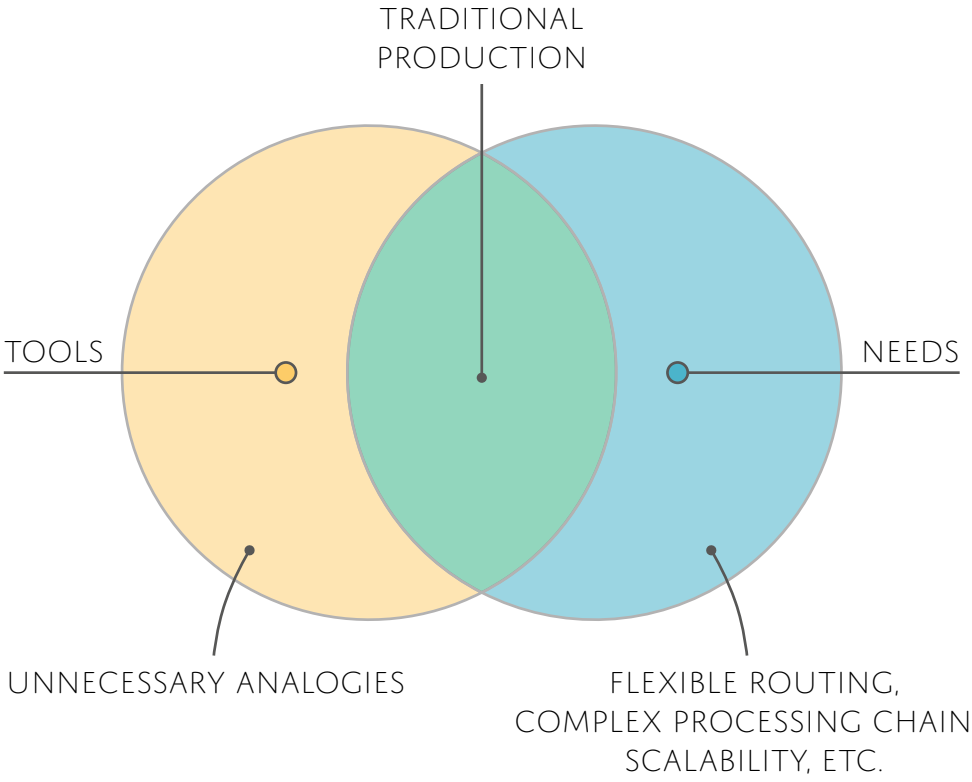


Figure 12: The relationship between the tools and the needs in a modern music production environment. The modern tools still match the traditional production methods more or less, but a shift in the needs is taking place. In an ideal situation, the two circles overlap each other completely.

From the standpoint of user interface design, the fact that a product enables the user to accomplish a task is not enough in itself. Norman and Wadia (2013: sec. 1, para. 1) phrased the problem appositely in their paper about touch and gesture-based systems: "Yes, getting the technology to work is hard, but the really hard part is getting the human-system interaction right, making it easy for people to use the systems." Although the users of digital audio workstations evidently manage to create great results, the emphasis in this thesis has been placed on inspecting *how* these results can be achieved.

Methods to modernise interface functions currently governed by the traditional paradigms are proposed in this Chapter, which involves weighing up familiarity on the one hand and improvements in usability on the other. Raskin (2000: 4–5) has discussed this dilemma:

> "Where real improvement can be achieved by making major changes, the interface designer must balance the legitimate use of familiar paradigms, which ease the learning process, against the enhanced usability that can be attained by abandoning them. In a situation of rapid turnover of personnel or the customer base, familiarity might be the better design choice. *Where most of the users' time will be spent in routine operation of the product and where learning is only a small part of the picture, designing for productivity—even if that requires retraining—is often the correct decision.*" (italics in the original)

A way to strike a balance between the two aspects is sought in this concept. Learning is an essential part in the use of a digital audio workstation: in addition to learning to accomplish tasks mechanically, the user needs to be able to use this information in creative ways. The learning process is therefore hardly trivial – even if the learnability of the system itself is good. This makes refreshing the paradigms especially challenging.

Observations about the track-based timeline paradigm

Current track-based timeline views are commendably flexible for several reasons. Firstly, the waveform-based visual editing is still remarkably functional despite the lack of substantial recent developments. Secondly, the impact the vertical and the horizontal zoom factors have on the versatility of the interface is significant: for example, zooming out allows arranging the project and zooming in enables very accurate cutting and splicing. Lastly, the interface can easily be adjusted to suit different scenarios, e.g. by overlaying automation data or increasing the size of particular tracks in relation to others. For productions that are based on linear time, the current track-based environments offer great possibilities.

There is room for improvement, however. For instance, tracks reserved for real-time processing, e.g. auxiliary tracks, may occasionally display only little information in relation to their size on the screen. Additionally, the need for specific track types can be questioned, and in fact, there have been some concepts for more

flexible tracks. For example, Bitwig (2013a) promises to include *hybrid tracks* in its upcoming digital audio workstation Bitwig Studio. These tracks enable the use of both audio and virtual instrument material on the same track (Bitwig, 2013b).

Conceptualising any fundamental changes to the timeline interface – to support, for example, probabilistic composition methods and other composition forms for which the linear timeline does not offer enough flexibility – goes substantially beyond the scope of this thesis. The focus of the concept presented in this text is on the mixing console paradigm, as the limitations of the analogue mixing console metaphor are viewed here as more acute than those of the timeline paradigm.

Observations about the mixing console paradigm

The current mixing view paradigm, used in many major digital audio workstations, is highly functional for procedures that were typical in the analogue environment. Simple mixing tasks, such as level adjustments, and traditional, channel-based audio processing, e.g. manipulating the spectrum with an equaliser and controlling the dynamic range with a compressor, are easy operations in practically any digital audio workstation. In many situations, however, the analogue console-like paradigm is inefficient or restrictive. For example, most current digital audio workstations share the following restrictions:

1 Specific routing and processing elements, namely insert effects and signal sends, have fixed signal chain positions, or there is only a choice of either pre-fader or post-fader positions;

2 Insert effect slots conceal the audio processing they contain – signal manipulations that are visually hidden may have vast audible effects;

3 Inspecting the signal flow visually in complex sessions, where multiple channels send signals to other channels, is difficult.

The inherent qualities of the mixing console metaphor cause these restrictions. First, a typical digital audio workstation channel strip relies on a fixed, highly analogous element structure, which leads to inflexible signal routing options. In addition, most of the elements have a static size. In other words, the current means to adjust the interface to match the needs of specific uses are insufficient.

There is an important distinction between the problems related to the track view and the issues of the mixing console view. The graphical track-based timeline interface is, in a sense, more modern: it utilises the capabilities of the current display devices by presenting a signal representation completely unavailable in the analogue domain, it allows flexible scaling, and it is arguably very simple to use in relation to the possibilities it offers. The mixing console view, on the other hand, is reminiscent of an analogue device, including its rigid structure. This difference seems logical considering the different backgrounds of the two paradigms (see the section 2.4 "Background of computer-based digital audio workstations").

The role of metering

The restrictions imposed by the firm interface structure prominent in current digital audio workstations also affect metering. For example, channel level meters are typically realised in mixing console views as more or less static-sized elements grouped together with the level faders, and the choice of signal chain position for the level measurement is usually very limited. Although this approach is, in a sense, more modern than the traditional, separate meter bridges featured in many analogue consoles, the inability to resize[19] or move the meters restricts the utilisation of the possibilities specific to graphical user interfaces.

What is more, the internal dynamic range of a digital audio workstation system is today immensely large; careful observation of channel meters is therefore, in many mixing situations, no longer necessary to avoid signal clipping. Consequently, reserving a significant portion of the display area for displaying the instantaneous channel output level in decibels relative to full scale (dBFS) is much less practical than it used to be.

The ability to compare the level of a channel with other channels is a feature of great importance, however, and the current meters are adequate tools for that purpose. Moreover, modern digital audio workstations offer scales that can be used as alternatives to the full scale, e.g. different forms of programme level measurements. In addition, metering can be used, for example, as a means to navigate to the right channel. The metering interface elements could nevertheless be expanded to include visualisations of signal spectra, for instance.

19 This restriction is common, but not universal. For example, Steinberg Cubase allows resizing the meters along with the faders (Steinberg Media Technologies, 2013b). Similarly, Ableton Live features resizable meters (Ableton, 2013).

4.1.2 Main aims of the interface concept

The user interface paradigms identified in the paradigm examination (see Chapter 3) cover an extensive range of distinct subjects. Consequently, treating them all goes beyond the scope of this interface concept. The primary purpose of this concept is therefore clearly defined: the proposed interface is a mixing environment that offers more flexible interface elements in comparison to the established analogue mixing console-based paradigm.

Fundamentally, the proposed interface concept is still comparable to the established mixing console-based design. However, instead of using rigid interface elements in fixed positions to constitute the interface, the concept consists of *process blocks*, i.e. abstractions of the traditional channel strip elements. This allows discarding some of the most restrictive analogies.

In terms of a complete digital audio workstation interface, the concept is proposed as a replacement for the mixing console view. The interface concept does not cover the functionality of the track-based timeline paradigm. However, the proposed structure is not dependent on the timeline view either: the concept is designed to enable the timeline tracks and the mixing view channels to be separated from each other. This provides a solid foundation for an audio workstation that is less dependent on the linear timeline and the track-based division of sound material, and yet supports the traditional division between the two structural interface paradigms. In other words, the tracks in the timeline view and the channels in the mixing view can be unlinked, and the channels can use virtually any other types of sources instead.

The conceptual aim of this interface concept is to restore the power of creativity for modern-day music producers. In this proposal, the established metaphorical interface structure used in the current digital audio workstations, which support the traditional sound production workflow, is considered insufficient. However, the traditional way of working is not disregarded in the proposal; on the contrary, the aim here is to expand the vocabulary instead of changing the way music is produced.

On a pragmatic level, the aim of this concept is to offer designs for central building blocks from which a concrete, modern digital audio workstation mixing interface can be built. Schematic abstractions are provided to demonstrate the interface and the discussed concepts. The focus is on the central interface elements and the

structure of the interface, but additionally, essentials of interaction are described. In other words, this text does not portray an implemented interface; instead, a concept for a modern interface structure is presented.

The interface concept is designed to support a variety of different input methods and display devices, particularly traditional desktop computers and current touchscreen devices. On the other hand, the features of the concept are not restricted to any specific device platform; the aim is to present highly scalable ideas that can be applied to a number of distinct systems. Proprietary platforms are therefore not discussed here.

This proposal describes the concept from the point of view of high-level user interface design. Consequently, discussion of the technical implementation goes mostly beyond the scope of this thesis. It is important to point out, however, that the scalability of the interface – a concept emphasised strongly in this proposal – would probably require special attention in the implementation phase.

Functionally, the bulk of the changes this concept would likely require to existing means of audio handling relates to signal routing. Some of the features of this concept might require significant modifications to the current methods, while others, provided the user interface is thoroughly separated from the audio processing, should be easier to implement. However, this interface concept proposes a front-end, a fundamental attribute of which is to offer streamlined access to the capabilities that are already largely available in current audio workstations through complex interaction.

4.1.3 Proposal

This concept offers an alternative for the traditional mixing console-based interface design. Three novel concepts differentiate this concept from the established paradigm:

1. Abstract *process blocks* allow processing and routing signals in a flexible manner;

2. Revealing the signal flow is possible with *routing inspector*, a modal view;

3. The interface is highly scalable.

The first of these concepts – the process block – is the most essential, and is the core of the interface. Shneiderman (1998: 74–75) described "the Eight Golden Rules of Interface Design", the first of which is to "Strive for consistency." This is one of the principal aspects of the process block structure: the interface is built from blocks, which do perform different functions, but which are considered equal from the standpoint of the user interface. In other words, channels can consist of virtually any blocks in any order.

Norman's model of interaction describes task execution (see the section 2.3 "Concepts of interaction and usability"). Norman (1998: 45–49) specified seven stages that occur when a task is performed, and divided these further into three aspects: forming a goal, executing an action, and evaluating the results. The process block concept supports efficient task execution even in scenarios where the user is unfamiliar with the system, e.g. during the initial learning phase: the system is based on a coherent set of elements and procedures, which allows the user to apply interaction patterns learned during a given task to many other tasks. Therefore, performing tasks on a trial-and-error basis is expected to lead to successfully completed tasks in many cases.

The second concept – the routing inspector – offers a way to see information about the signal flow at a glance. In tradition mixing console views, the common way to inspect the "horizontal" signal flow – that is to say, routings from one channel to another – is to read the output names from the output or send designators. This is arguably an insufficient method to comprehend the signal flow structure quickly, especially when working with a complex session. As this interface concept includes even more signal routing flexibilities in comparison to the traditional mixing console paradigm, providing a visual way to inspect the signal flow is crucial.

The third concept – the scalability – is important for two specific reasons. Firstly, scalable interface design allows adapting the interface to different scenarios and specific uses – the individual process blocks can be scaled, which allows displaying more channels with less details as well as less channels with more details. Secondly, the interface concept is designed to support different input devices and display devices, namely traditional desktop systems and touchscreen devices of varying size.

4.1.4 Presentation

The illustrations in this Chapter are not supplementary to the concept; rather, the concept consists of the figures. In order to focus on the central aspects of the interface, figures are utilised to demonstrate specific concepts instead of portraying the entire user interface in exhaustive detail. Although static figures are capable of illustrating only certain aspects of the interface, expandability and scalability are central to the proposed design. Thus, the presented techniques can be extrapolated to situations not discussed here.

The interface structures and functions are represented as wireframes and schematised mock-ups. Properties of visual style are not part of this proposal. Consequently, the concept has been developed in grayscale, which places emphasis on the proposed concepts instead of attributes of graphic design. In a concrete implementation or a high-fidelity prototype, the central aspects presented here should be accentuated with, for example, carefully selected colour values and refined shapes.

4.2 Interface structure

The process block interface concept is based on columns. A column consists of an array of process blocks, and the number of blocks a column can contain is unrestricted. The Gestalt laws of grouping (see the section 2.3 "Concepts of interaction and usability") have been used as significant design tools in the development of the interface concept; in fact, the process block structure is essentially based on carefully defined groupings.

Elements related to a single function, e.g. a destination selector and a level control for a signal send, are fused into entities, and these entities, the process blocks, are further combined into columns. The block column is the most central part of a channel[20], i.e. an entity consisting of one or more related signal paths. The number of channels can be changed without restrictions. The basic structure of the block-based mixing interface is illustrated in Figure 13.

20 See 2.1 "Terminology".

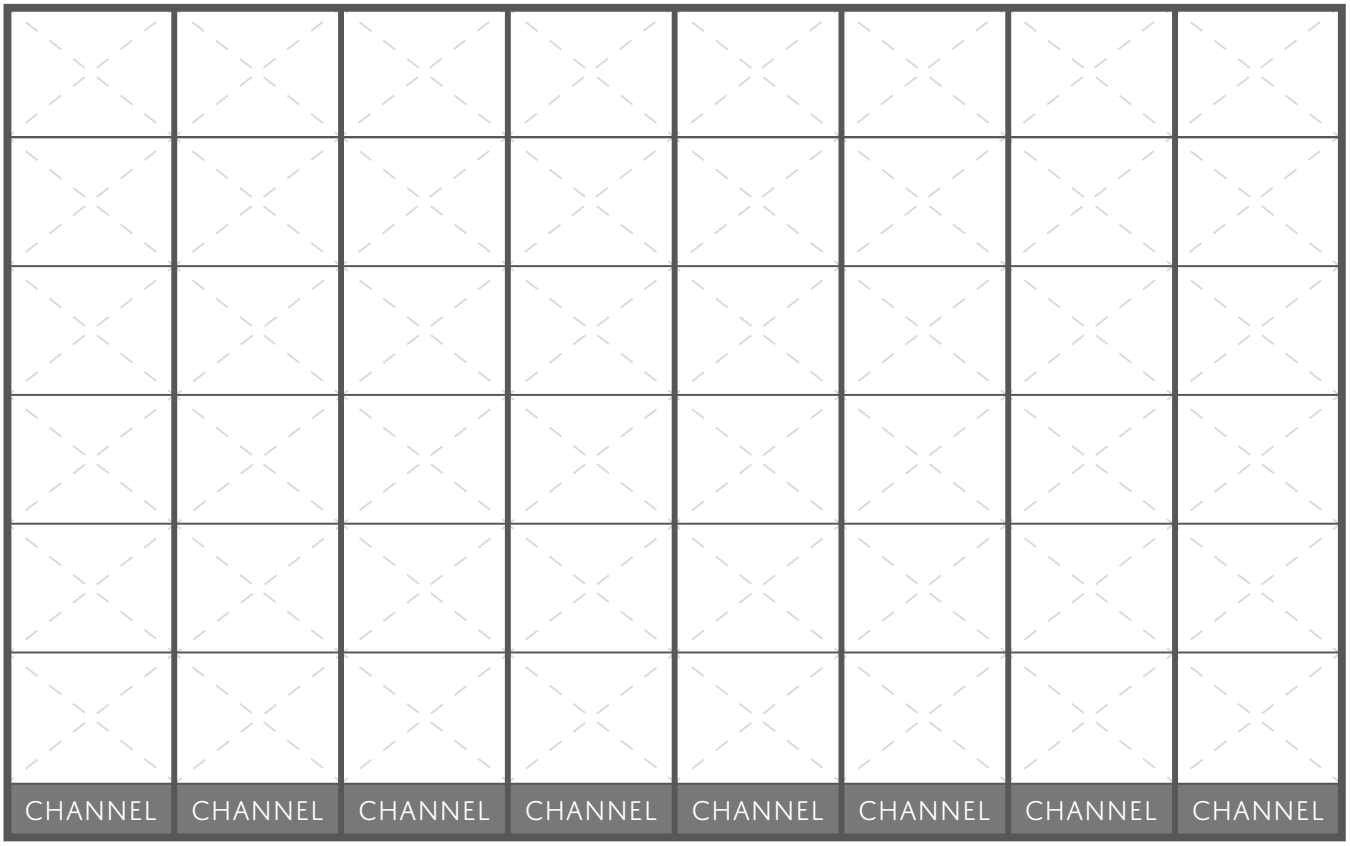


Figure 13: The fundamental process block interface structure consisting of vertical chains of blocks. The number of channels and the number of blocks contained in each column can be changed without restrictions.

The basic concept of the process block is described in the subsection 4.2.1 "Process block-based mixing". The subsection 4.2.2 "Signal flow" presents the fundamental approach to signal flow specific to the block interface.

4.2.1 Process block-based mixing

The wireframe interface structure illustrated in Figure 13 does not portray any signal flow elements; to initiate the signal flow, process blocks need to be added to the boxes. A bare minimum setup to allow signals to travel through the system would consist of source blocks and output blocks. New channels could also be initialised with, for example, a source block, a metering block, and an output block.

Thus far, the basic principle is quite similar to existing console-based interfaces, but the main difference between the console structure and the column-based block structure becomes evident when assessing specific channel elements. The

common processing and routing elements available in current digital audio workstation channels offer many possibilities for processing the audio. However, the categorisation of these elements restricts the versatility of the mixing environment: for example, channel effects are typically pre-sends, and the user cannot change this behaviour. The mixing view structure presented here changes the focus from the specific processing and routing elements to generalised *processes*, which can be placed in virtually any order. Figure 14 illustrates the core element functions abstracted from a schematised traditional channel strip representation.

The abstract channel elements provide the basis for the process blocks; all routing, processing, and monitoring chains are handled in this concept with the process blocks. The blocks are categorised into four categories according to their functionality. However, the categorisation merely describes the primary element functions, and does not restrict the signal chain positions of the elements. The core of the interface consists of:

1. Routing blocks (a source block, a send block, and an output block);

2. Gain blocks (a level block and a panning[21] block);

3. Meter blocks (different types of meters);

4. Effect blocks (insert-like signal processors).

The mixing environment structure is not limited to just the blocks mentioned above, however. The process block-based structure is highly flexible, and from the user interface standpoint, new specialised blocks can be added with ease.

Different process blocks still benefit from different renditions. The purpose of some traditional channel elements, namely the group designator and the automation mode selector, is essentially to offer a means to select a value for a property (and to display the value). Although the aim in the process block structure is to allow the user to build the mixer structure from individual, detached building blocks, offering separate blocks for such functions would result in unnecessary clutter and complicated access to these properties. Moreover, these functions are in essence channel settings, not elements that are placed on the signal path.

21 Panning is considered here a gain operation for the simplicity of categorisation; the basis for this categorisation is that panning can be thought as a simplified control for simultaneous adjustment of multiple channel levels.

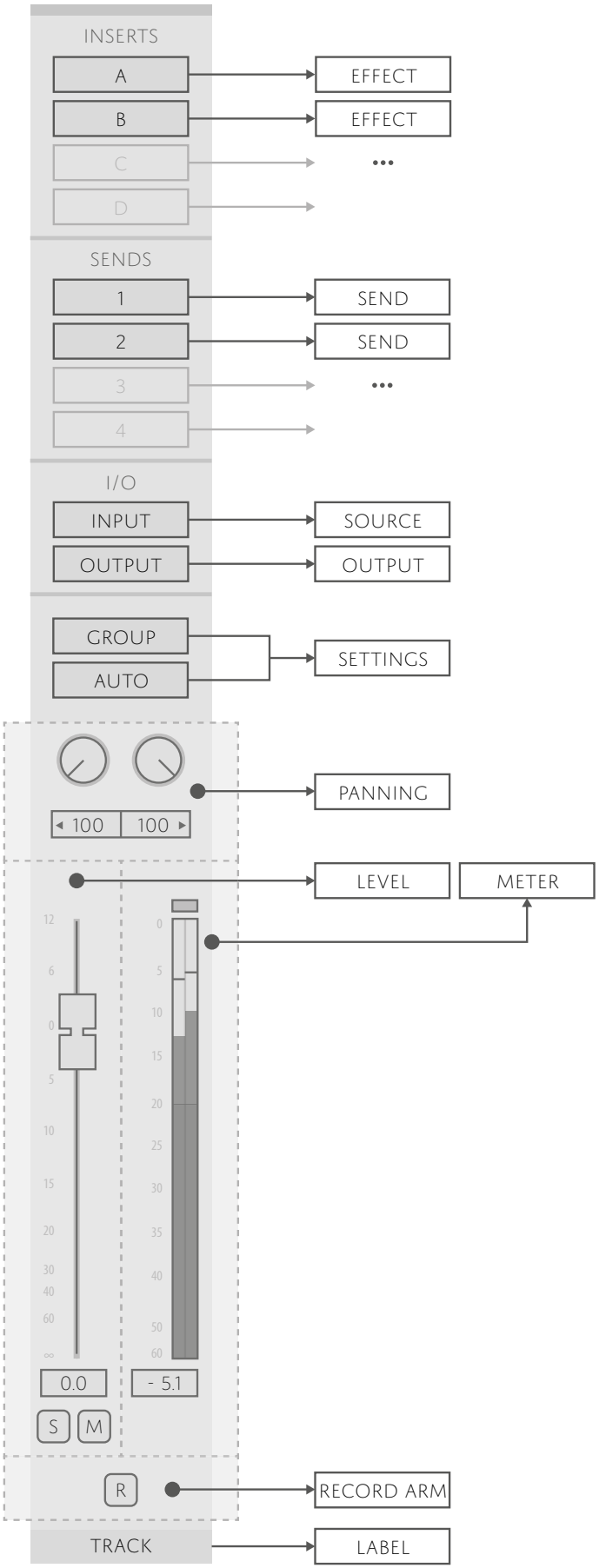


Figure 14: The conversion of the essential channel strip elements into abstract stereotypes.

Supplementary properties such as the group selectors and the automation controls are grouped in the process block interface into settings elements, which are separated from the main block columns. Although this may seem to contradict the generalised block-based concept, this approach in fact allows the block columns to retain their unambiguous signal flow. The block interface is illustrated in Figure 15 with the separate settings elements.

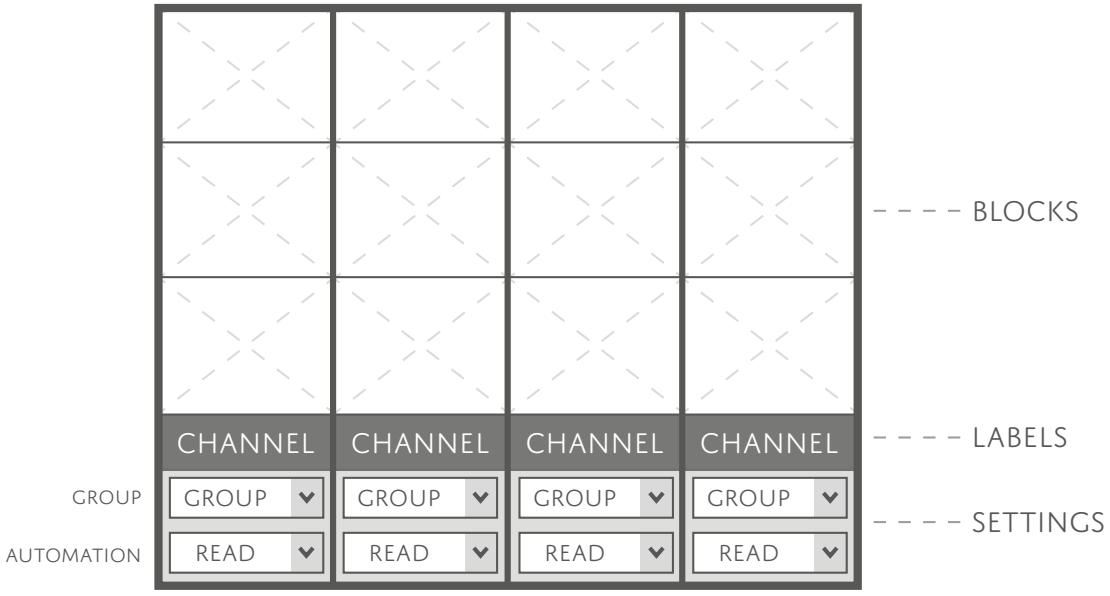


Figure 15: The process block interface with the settings elements. Channel settings are separated from the main block area: the channel labels act as dividers between the block columns and the settings.

Selecting a source or an output for a channel is comparable to assigning a group or selecting an automation mode from the standpoint of the user interface. However, the channel source and the channel output relate very directly to the signal path, and hence these functions are implemented as individual process blocks in the block interface.

4.2.2 Signal flow

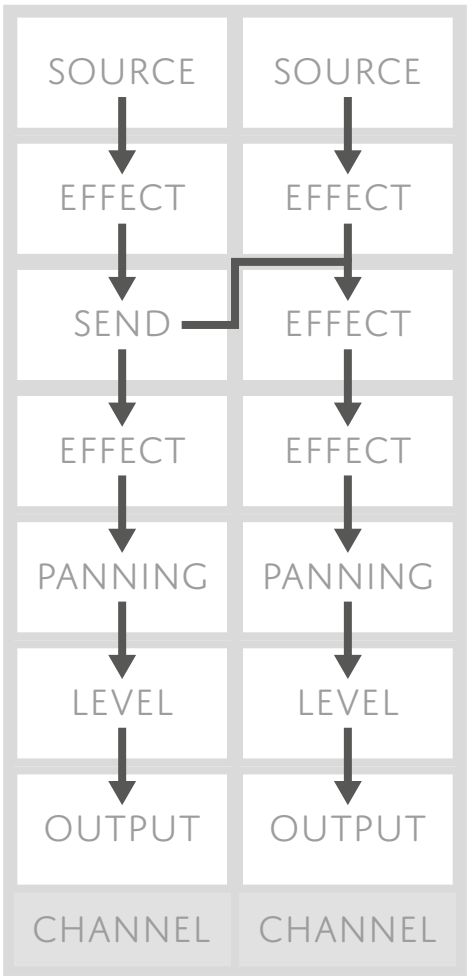
The signal flow within a channel is very simple in the process block interface: the signal flows vertically, starting from the topmost block. Whereas in traditional channel strips certain elements, e.g. input and output selectors, commonly have fixed positions that do not necessarily correspond to the actual signal chain, the flow in the process block interface is user-definable and unambiguous: there is a one-to-one relation between the order of the blocks and the channel's signal chain.

Deviations from the main flow

A signal flow structure based solely on separate channels would be insufficient. For example, using a single reverberation device for multiple channels is often practical. A means to send a signal from a channel to another is therefore included in the process block concept.

A *send block* can be used to deviate the signal from the default signal path. The send block is basically similar to the output block, but there is one important distinction between the two. The output block is a terminal: blocks cannot be placed post output. The send block, on the other hand, allows signals to pass; signals can be, for example, processed with effects and outputted to another output post send.

Two main aspects set the send blocks apart from the send structures of the current digital audio workstations. Firstly, a send block can be situated in virtually any signal chain position, e.g. pre effects, or post the first channel effect but pre the second effect. Therefore, user has fine-grained control of the sent signal.



Secondly, similar flexibility is offered in the receiving end: signals can be sent not only to the top of the receiving channel's block column, but also to any other position. In addition, there are no specific auxiliary channels: any channel can be used as a target for a signal send. Explanatory signal paths are illustrated in Figure 16.

The send blocks also enable the creation of feedback loops and feed-forward paths. In other words, signals can be sent to a destination that is contained within the same channel. In an actual implementation, however, providing a safeguard switch for enabling and disabling the feature would be essential in order to prevent accidental loops. The feature should be disabled initially, and enabling it should prompt a clear confirmation dialogue.

Figure 16: Signal flow in the process block interface. Signals flow within channels from top to bottom, and send blocks can be used freely for creating additional signal paths.

4.3 Principal features

The most fundamental features, functions, and attributes of the process block interface are described in this section. The specific ways to initiate the signal flow within a channel are presented in the subsection 4.3.1 "Recording and initiating the signal flow". The effect handling in the process block interface is presented in the subsection 4.3.2 "Effect encapsulation", and the subsection 4.3.3 "Level control and metering" discusses the aspects of controlling and observing the channel levels.

Resizing of the interface elements, one of the most fundamental features of the interface concept, is discussed in the subsection 4.3.4 "Block resizing". A structural feature to restrain the consequent effects of the flexible resizing is described in the subsection 4.3.5 "Pinning". Lastly, the subsection 4.3.6 "Flow representation" presents a specific view mode that provides a visual means to inspect the signal paths.

4.3.1 Recording and initiating the signal flow

Recording is integral to audio production, and consequently, recording cannot be disregarded here even though this interface concept relates primarily to mixing. Typical channel representations in the current mixing console views contain two interface element types essential to recording: the input selector and the "record enable" or "record arm" button which enables recording the signal from the selected input to the particular track.

In the process block interface, the two recording controls are combined into a single element: the *source block*. In essence, the functionality of the source block corresponds to the established way of recording in digital audio workstations. However, a distinction between a track and a channel is essential to the source block: "channels" are the main entities constituting the process block interface, whereas "tracks" are entities accessed through a timeline view.

An "arm" button in the source block routes the signal from the selected input to the selected track, and hence enables recording the input to the track (see Figure 17). If a track is unarmed (see Figure 18), the source block will effectively ignore the input, and the track contents will be used for playback. In other words, tracks are considered sources in the process block interface. A channel can either use the audio regions already on a track as an input or monitor a live input when a track is armed.

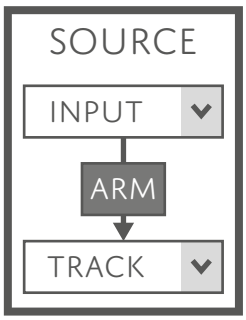


Figure 17: The source block in "record" mode.

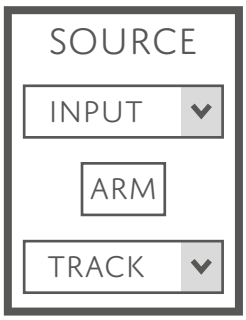


Figure 18: The source block in "playback" mode.

Placing a source block on a channel is optional: channels reserved for parallel effects, for instance, can omit source blocks altogether. When placed on a channel, the source block follows the main vertical signal flow: the source block has to be the topmost element of the channel, as it is the initial source for the channel. Within a source block, the input is displayed above the track, which reflects the signal flow between the two. Input monitoring could also be detached from the record arming for example by adding a separate "Input Monitor" button to the source block.

In addition to the visual signal path representation, the source block has another distinct advantage over the established approach: the same track can be used as a source for multiple channels. This enables the creation of multiple processing variants of the same original material without duplicates of the original track or signals routed with sends. Assigning different versions of an instrument for different sections of a work, for example, is therefore only a matter of automating the channel selection. Alternatively, the channels can be used simultaneously for parallel processing.

The flexibility offered by the source block also entails some usability challenges. Firstly, when a new track is created, a corresponding channel should be added automatically – at least as the default behaviour of the system. Requiring the user to create a channel manually every time a new track is added is unjustifiable. Secondly, the source block concept allows routing multiple different inputs to a single track. This could be turned into a significant feature, however: the signals could be summed or layered as alternatives for each other – offering a simple way to compare multiple microphones, for instance.

4.3.2 Effect encapsulation

Inspecting the current channel strip elements reveals that while some of them – the level fader, for example – are quite functional, others can be improved. In particular, effect handling has room for significant improvements. Modern effect plug-ins have demonstrated novel ways to communicate useful, readily interpretable information about the processed signals to the user. For example, FabFilter Pro-C compressor plug-in displays the signal compression as scrolling representations of the input signal, the output signal, and the gain reduction (FabFilter, 2013). The interface of Pro-C is shown in Figure 19 (FabFilter, 2013).

Figure 19: FabFilter Pro-C compressor plug-in displays the input signal, the output signal, and the gain reduction in relation to time (FabFilter, 2013). Screen captured by Petri Myllys. Used with permission of FabFilter.

The *effect process blocks* allow visualising the processing that occurs inside the encapsulated effects. Instead of displaying only the name of the effect, an effect block can display information about the current state of the effect. Therefore, observing the influence of an effect does not require opening a separate plug-in window.

The aim is not, however, to create a completely self-contained mixing view – detailed effect control is handled with separate effect windows in order to keep the information density in the main block interface appropriate. Explanatory process blocks, namely equaliser and compressor blocks, are shown in Figure 20. The concept of time-dependent compression visualisation is explained in Figure 21.

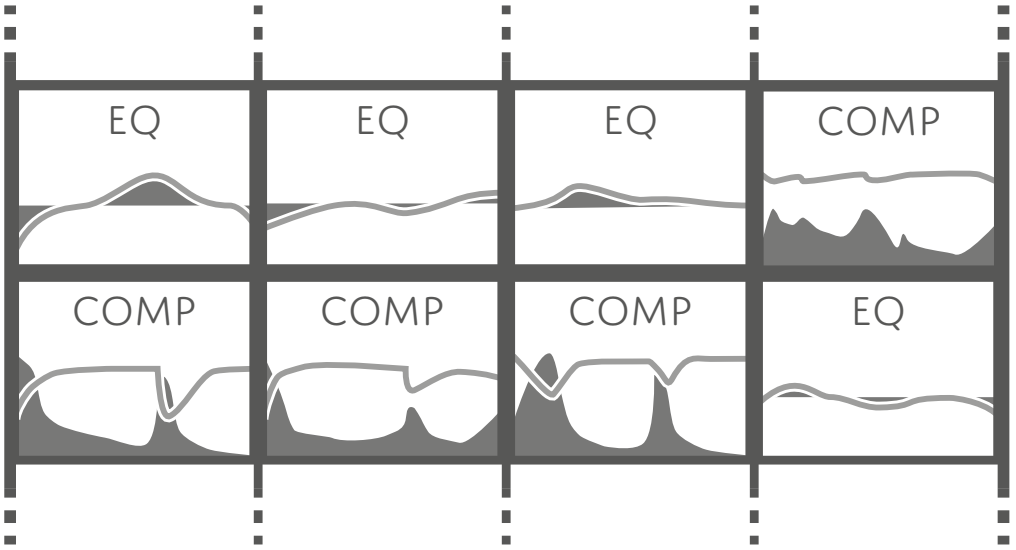


Figure 20: Two rows of process blocks in a four-channel session. Equaliser blocks display miniature curves; compressor blocks display the input level and the amount of compression in relation to time.

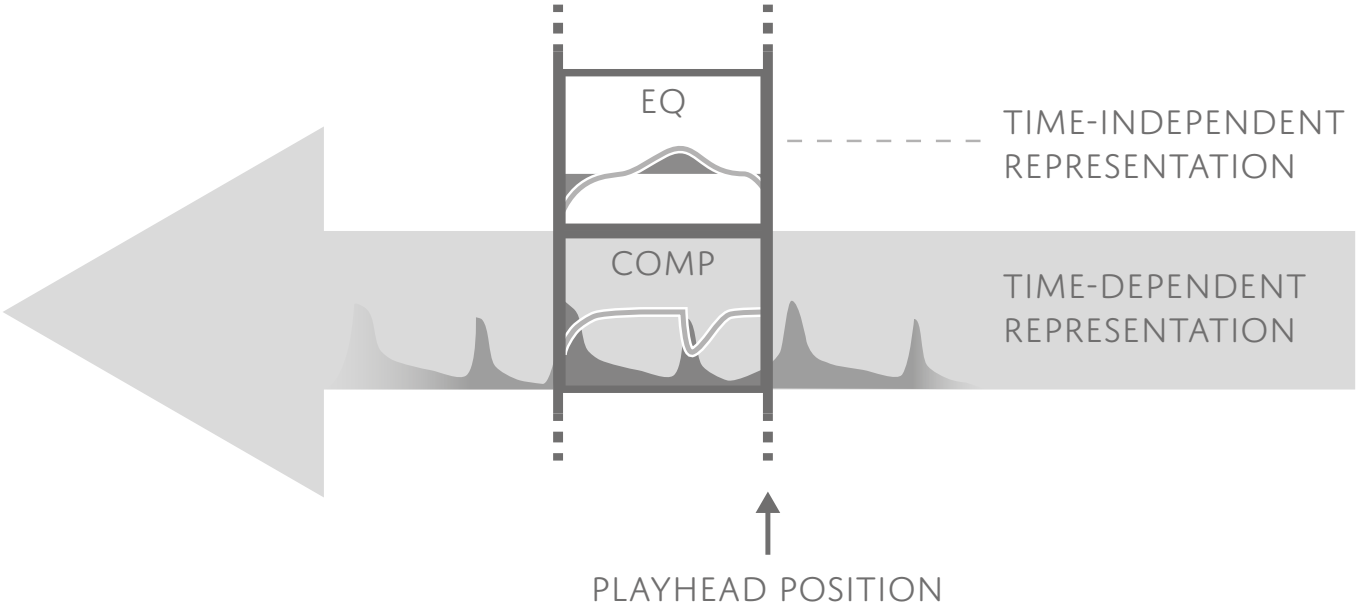


Figure 21: Time-dependent compression visualisation. The information displayed in a compressor block scrolls across the element according to the transport of the workstation. The block provides a viewport through which the signal can be inspected a small part at a time, resembling the way the track-based timeline views work.

The described approach introduces a specific challenge from the technical perspective: the visualisations would require proprietary plug-ins, or alternatively an application programming interface and corresponding implementations by the plug-in developers. However, including basic effect devices as a bundled part of a digital audio workstation is common; integrating the key effect devices this way would allow taking advantage of the visualisations.

The user may nevertheless want to use a certain third party plug-in which cannot visualise its effect. This should be appreciated. In such scenarios, the process block element can be repurposed: blocks representing third party plug-ins can offer access to certain plug-in parameters without opening the plug-in window. This kind of an approach is used for the third party plug-in handling in Ableton Live (Ableton, 2013).

4.3.3 Level control and metering

The common way to display level meters in digital audio workstations is to group them with the level faders. This is sensible: the two elements are closely related, and situating the meters next to the faders results in a compact yet organised layout. In the process block structure, however, the two elements are separate. On one hand, this allows superior flexibility, as the elements can be used individually in any signal chain positions. On the other hand, the flexibility creates challenges with regard to the element layout.

Displaying a single, large level fader and a single meter per channel is often sufficient, however. In such situations, separating the two elements and presenting them as individual process blocks would be inappropriate: display area would be consumed inefficiently, and the general usefulness of the approach could easily be questioned. Therefore, displaying the fader and the meter in the established way, side by side, is possible in the process block interface.

This approach is practical, but it also makes the signal flow representation ambiguous, as two blocks share the same vertical position. Besides, the flow direction in this situation cannot be fixed: an option to select the order of the elements in terms of the signal flow is necessary, as both the pre-fader metering and the post-fader metering modes are useful (see the section 3.4 “Metering”).

In the process block concept, a switch is included in between the two blocks for changing the flow direction. In addition to offering quick access to the option, the control also provides constant visual confirmation of the current pre–post mode status. The controls of all channels can be linked to control the metering mode globally. A level block and a meter block are shown side by side in Figure 22.

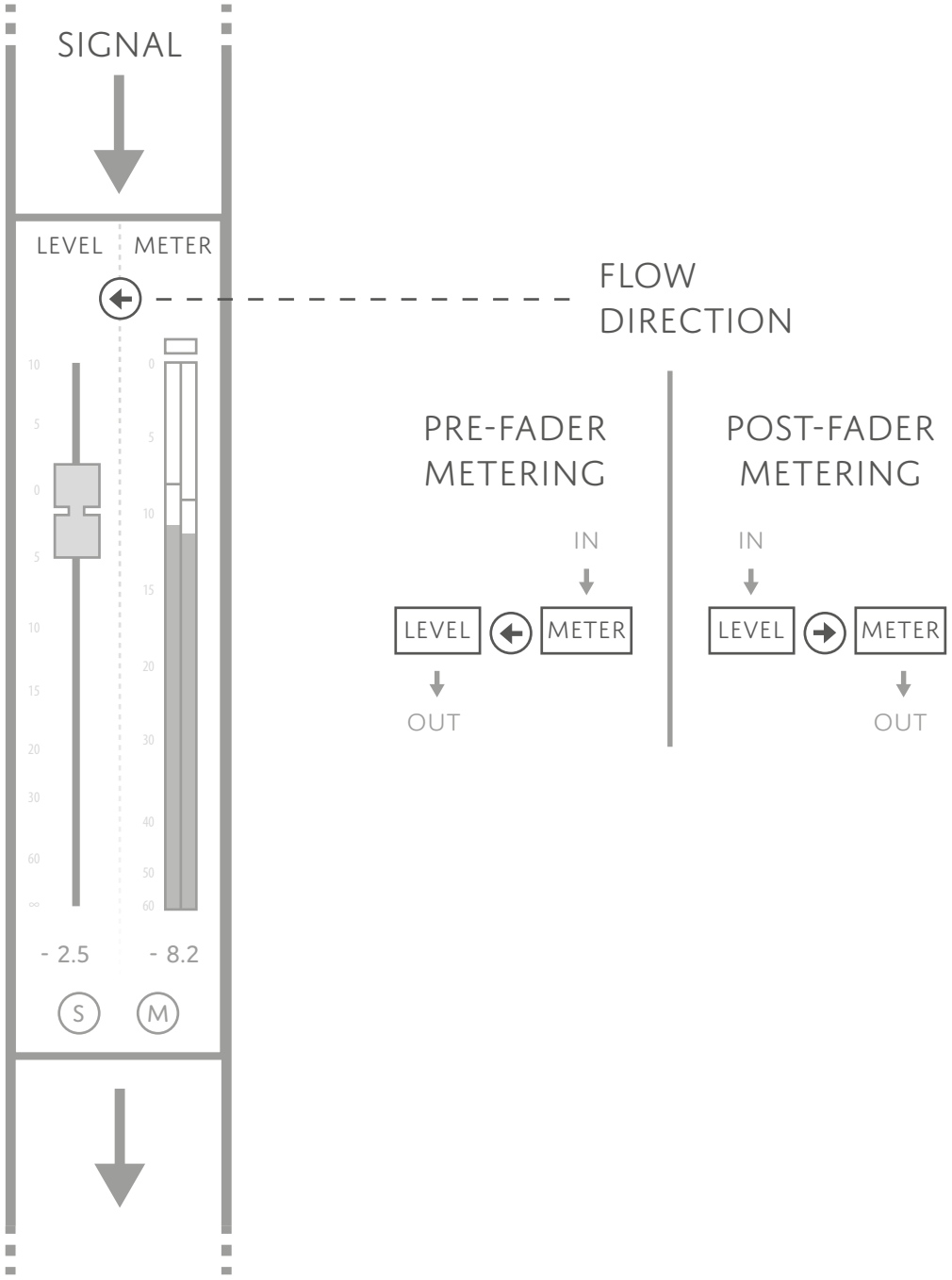


Figure 22: The level block and the meter block situated side by side. The order of these elements can be controlled with a flow direction button which represents the currently selected direction. In essence, the control is a pre-fader–post-fader switch for the meter. The direction can also be changed globally for all channels.

4.3.4 Block resizing

Different sound production tasks require distinct software functions. For example, quick and reliable monitoring is arguably the paramount aspect during recording, whereas the mixing phase requires, for instance, flexible processing options. Moreover, different genres benefit from different features, and even the user preferences may vary. To meet the diverse requirements, the elements in the process block structure are resizable.

Current digital audio workstation mixing console views feature some resizable elements, but this functionality is typically quite limited. For example, the send rows in Avid Pro Tools 11 can be expanded individually to display miniature level faders (Avid Technology, 2013e), but comparable functionality is not available for elements such as effects (Avid Technology, 2013c).

Resizing the mixer horizontally, i.e. adjusting the width of the channel strips, is typically possible in current mixing console views. However, changing the width has commonly very little effect on the functionality of the strips. The recent versions of Steinberg Cubase are notable exceptions to the general rule of restricted resizing. *MixConsole*, introduced in Cubase 7, offers means to resize the mixer both horizontally and vertically, and in addition, certain parts of the mixer can be resized individually (Steinberg Media Technologies, 2013b).

The process block interface consists of a user-determined combination of blocks. A block inessential in one session or situation may be of great importance in another scenario. For example, during a recording session, source blocks are most likely crucial, and additionally, send blocks could be enlarged for easier cue[22] mixing (see Figure 23). When mixing, however, the effect blocks and the main level controls may be of greater importance (see Figure 24). To cater for different situations with distinct requirements, all process blocks can be resized.

Although Figures 23 and 24 illustrate vertical resizing of the blocks, resizing the interface horizontally is also possible. This enables resizing the interface as a whole. Moreover, the block interface concept does not restrict the capability to few predetermined channel widths, which is commonly the case in the current digital audio workstations. Instead, the blocks can rearrange their contents according to the available vertical and horizontal space.

22 A specific monitor mix for the artist.

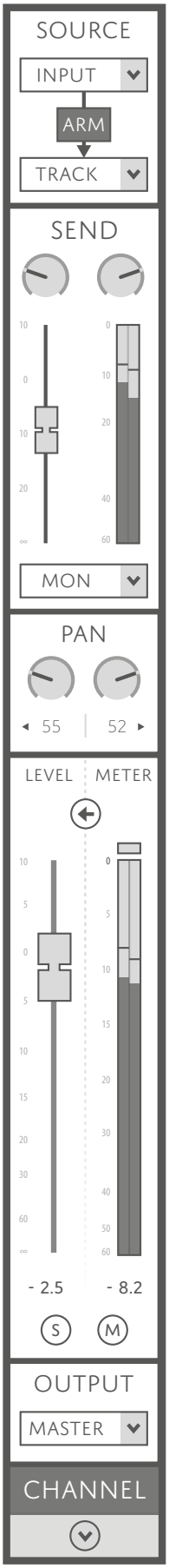


Figure 23: A block column for a recording session. The focus is on the source block, the send block, the level block, and the meter block. The channel settings are minimised, and the main meter is in pre-fader mode.

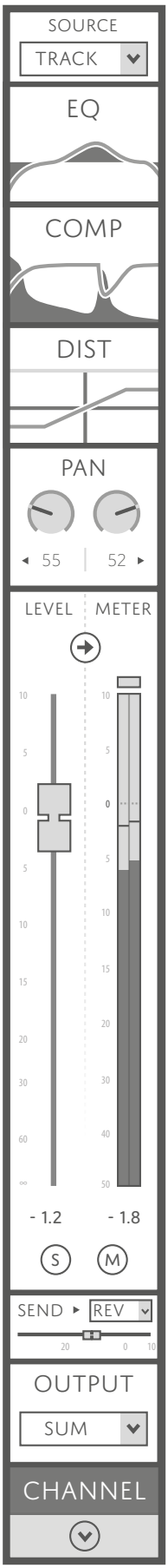


Figure 24: A block column for a mixing task. The focus is on the effect blocks, the level block, and the meter block. The channel settings are minimised, and small-sized blocks are used for controlling the source and the reverb send. The meter is in post-fader mode.

4.3.5 Pinning

Employing flexible and resizable process blocks has one specifically noteworthy consequence: similar elements do not reside adjacent to each other inevitably. For example, if the number of blocks placed pre the fader element differs from channel to channel, the faders are no longer vertically aligned. Similarly, using different-sized blocks on different channels easily leads to unwieldy results.

The problem related to adjusting the elements without any restriction is illustrated in Figure 25. The issue is, in fact, also visible in the Figures 23 and 24, if the two figures are considered two channels of the same session: comparing the fader positions or the values displayed by the level meters is practically impossible.

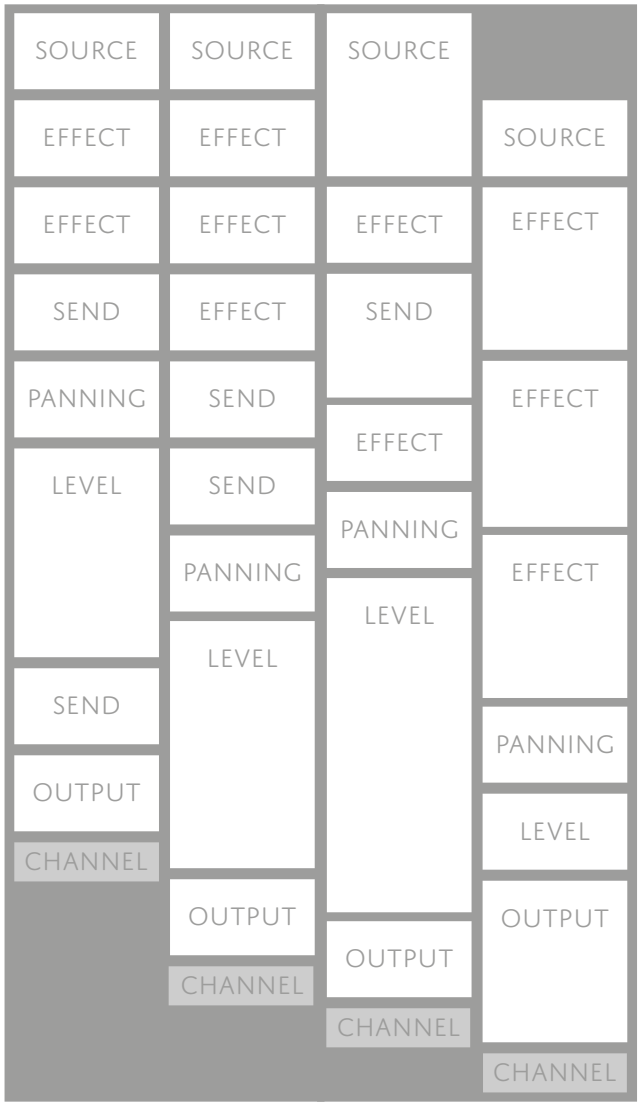


Figure 25: A schematised process block structure without "pinning". Adjusting the blocks without restrictions leads quickly to poor comparability across the channels.

To offer both the flexibility of the resizable process blocks and the structural integrity of the established mixing console paradigm, a balance between the two extremes will have to be struck. The method proposed here is in this concept termed *pinning*. In short, specific pinned rows can be inserted; blocks that are placed on a particular pinned row remain vertically aligned and similar in height. A configuration with pinned rows is illustrated in Figure 26.

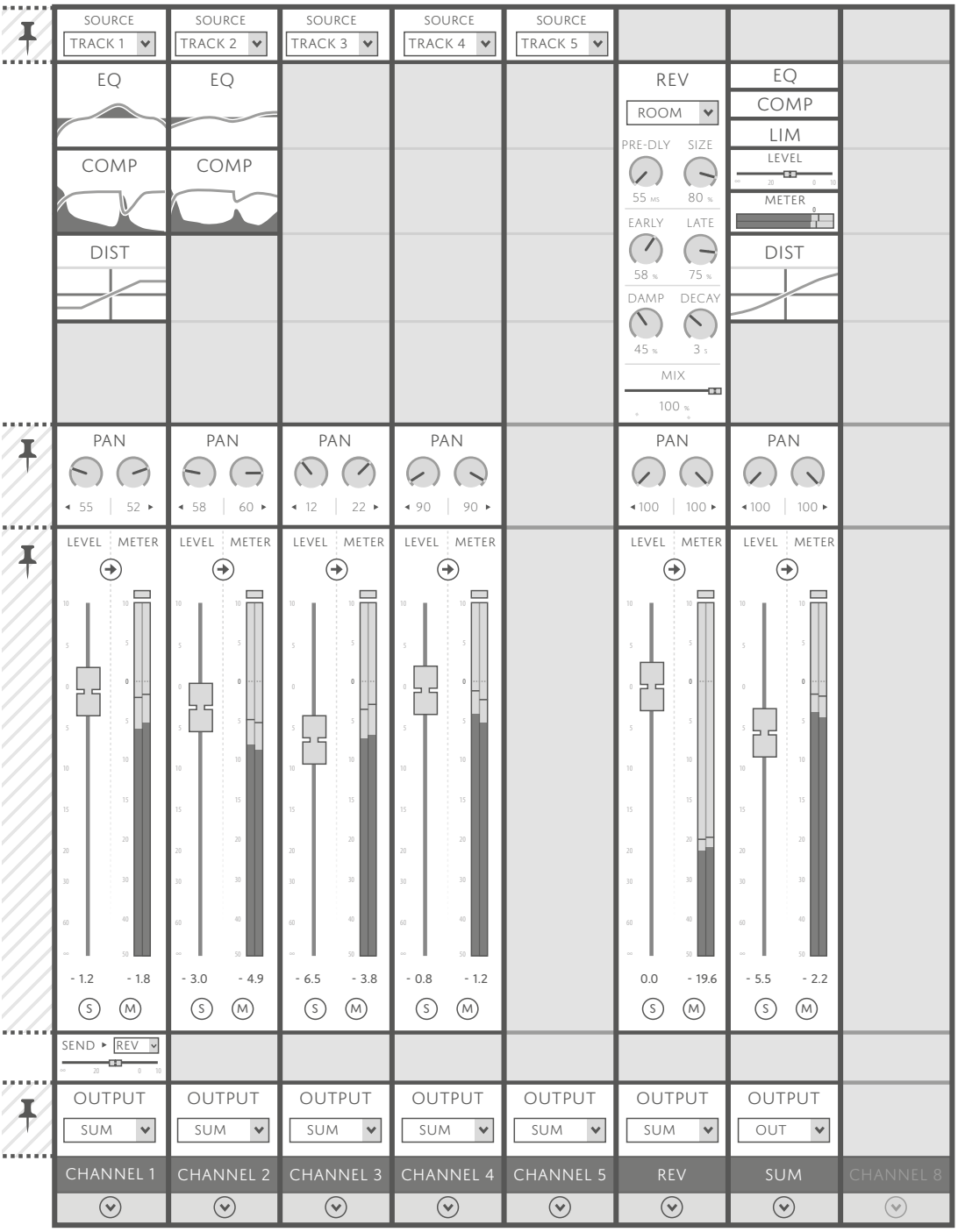


Figure 26: The process block interface incorporating four pinned rows. Blocks placed on a pinned row remain vertically aligned. The label row is pinned by default.

The addition of process blocks between pinned blocks remains unrestricted; a channel can still consist of any number of process blocks. If a column appears to have insufficient space between two pinned rows, adding a new block or enlarging an existing one will cause blocks on all channels to move correspondingly. Channel labels do not require pinning: the label row is pinned by default. The process of adding a block between pinned rows is shown in Figure 27.

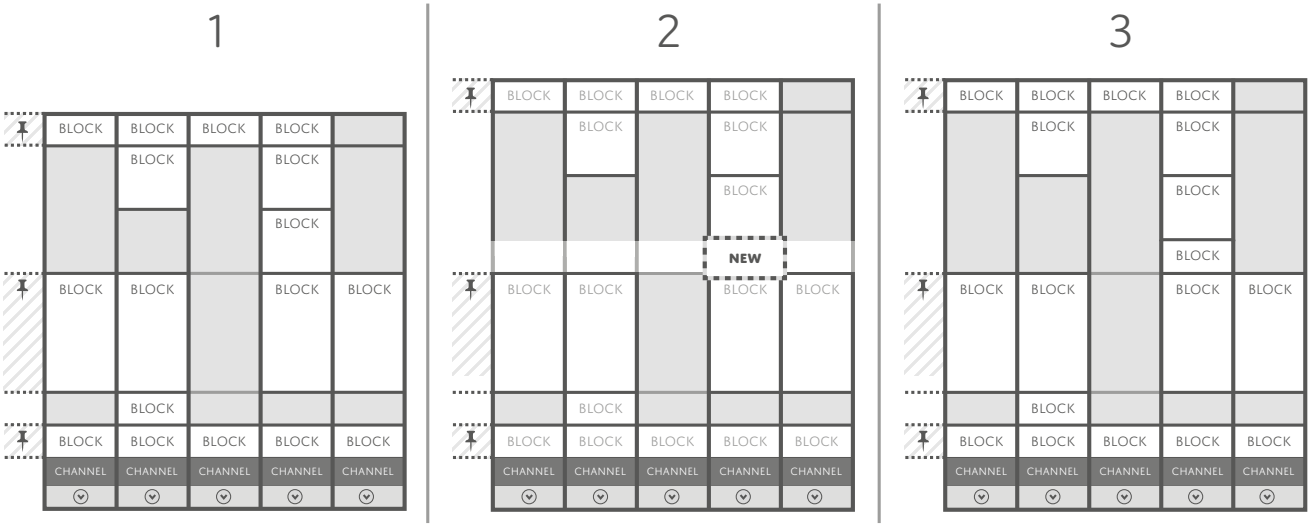


Figure 27: Adding a process block between pinned rows. The step 1 shows the initial situation. The step 2 demonstrates the interface state during the addition of the element. The step 3 shows the resulting configuration.

4.3.6 Flow representation

As discussed in the section 4.2 "Interface structure", the default signal path within a block column is straightforward. However, the process block interface also supports routing the signals in various ways. This demands sufficient methods for inspecting the signal flow.

Concepts of dataflow

Properties of visual dataflow programming provide the basis for the *routing inspector* of the process block interface. The routing inspector is a modal[23] view that allows inspecting the signal flow of the mixing session visually. The theory and the history of dataflow programming are not discussed here, however; these subjects go beyond the scope of this thesis, and the interest here is to present

23 See 2.1 "Terminology".

the fundamental concept of the inspector mode. The development of dataflow programming languages has been covered by, e.g. Johnston, Hanna, and Millar (2004).

Cycling '74 Max (Cycling '74, 2013a) and the open sourced Pure Data (Pd community, 2013) are prominent examples of modern dataflow programming languages – or dataflow software – that are very directly related to digital audio. Max, for example, incorporates MSP objects designed for audio processing (Cycling '74, 2013a).

Cycling '74 (2013b) describes Max in the following way: "You can arrange boxes on a canvas and connect them together to create, experiment, and play." MSP components and the visual dataflow programming paradigm allow the visual creation of, for example, virtual instruments and signal processing systems. The use of a dataflow programming software is essentially based on routing – or patching – the output of a box to the input of another box visually. A simple Max patch, i.e. a project file in Max, is shown in Figure 28 (Cycling '74, 2013a).

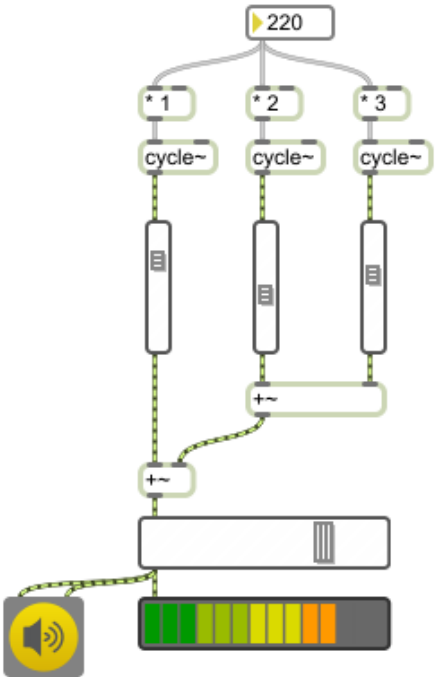


Figure 28: A patch in Cycling '74 Max dataflow programming software (Cycling '74, 2013a). Screen captured by Petri Myllys. Used with permission of Cycling '74.

Comparable routing systems are also featured in other types of audio tools, e.g. Buzz (Tammelin, 2013), Sensomusic Usine Hollyhock (Sensomusic, 2013), and Plogue Bidule (Plogue Art et Technologie, 2013). Additionally, Metric Halo audio interfaces incorporate a graph-based routing user interface as a part of a +DSP

expansion (Metric Halo, 2013). In fact, modular concepts have been used even in a digital audio workstation: Apple Logic Pro includes *Environment*, which is a dataflow-like view, albeit designed for MIDI routing (Apple, 2013c: 755).

The dataflow programming paradigm is, however, overly flexible for many tasks commonly associated with the digital audio workstations. For example, dataflow systems require managing the object arrangement manually, at least to some extent, which impedes the quick and easy execution of typical recording and mixing tasks. Moreover, adding new objects to a patch typically requires separate object creation and patching operations. However, the patch cable-based interfaces display the connections between the elements, which is an important distinction between such interfaces and the mixing console paradigm.

Routing inspector

Representing connections between multiple elements and allowing the system to be controlled simultaneously is not trivial. Visual dataflow programming environments demonstrate this problem: displaying the connections between the elements in addition to the modifiable elements themselves can quickly lead to unwieldy "patches" that, unless carefully managed, become very difficult to comprehend.

Therefore, this approach is not used in the process block concept. Instead, the routings are inspected using a specific *routing inspector* mode. The routing inspector is essentially a view filter. Activating the routing inspector and selecting a particular block displays a *routing indicator line* between the selected block and its destination. The destination can also be changed by dragging the end point of the routing indicator line to a new location. Deactivating the routing inspector hides the routing indicator line, and therefore prevents it from concealing the basic block interface.

The process block interface is illustrated with the routing inspector activated in Figure 29, and Figure 30 illustrates the same interface configuration with the routing inspector disabled. Scenarios in which this mode is practical include situations where information about the exact destination of, for example, a send is required. As only the connection from the selected element is shown, the amount of information displayed remains reasonable.

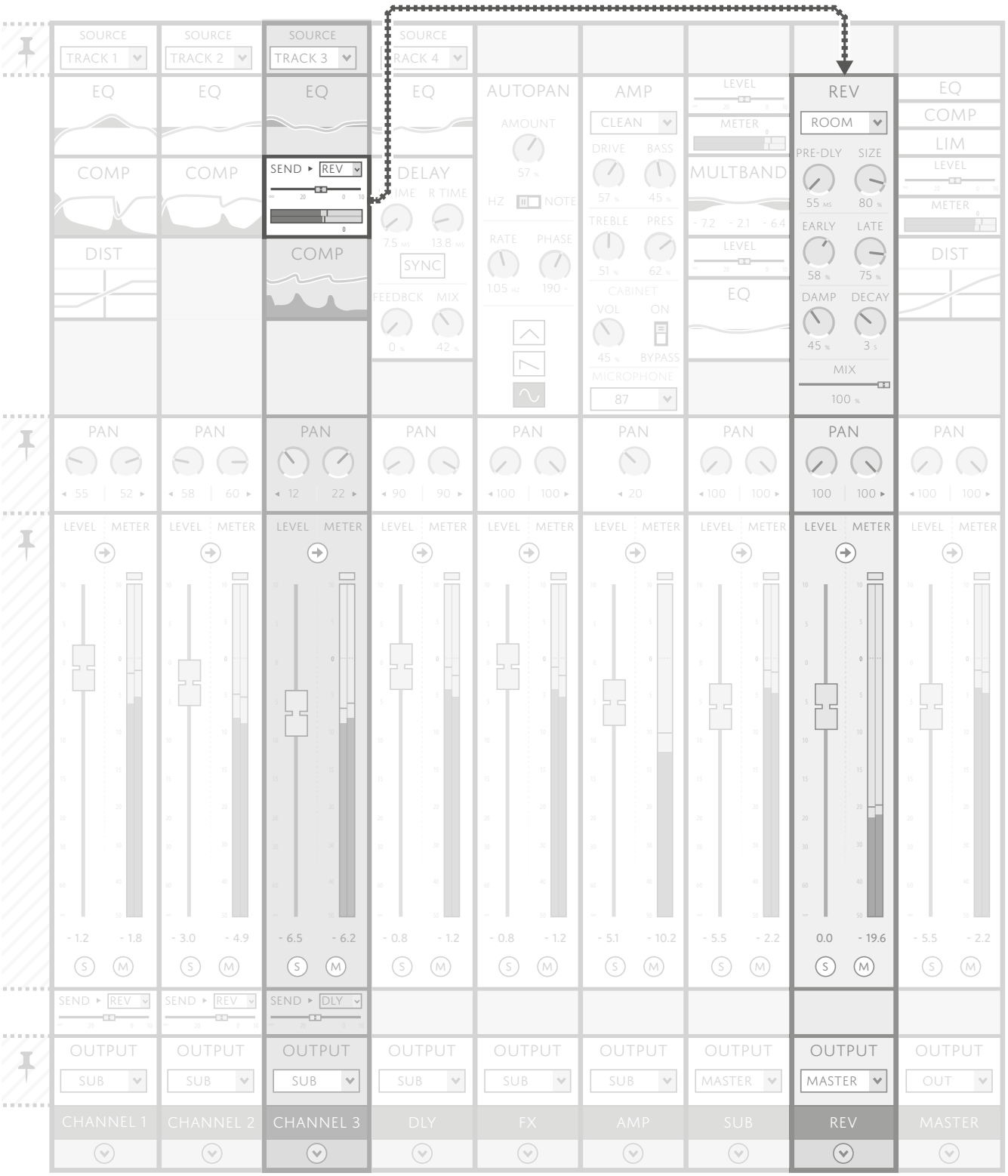


Figure 29: Routing inspector in use. The send block on the CHANNEL 3 is selected, and a routing indicator line from the send block to its destination, the top of the REV channel, is displayed.

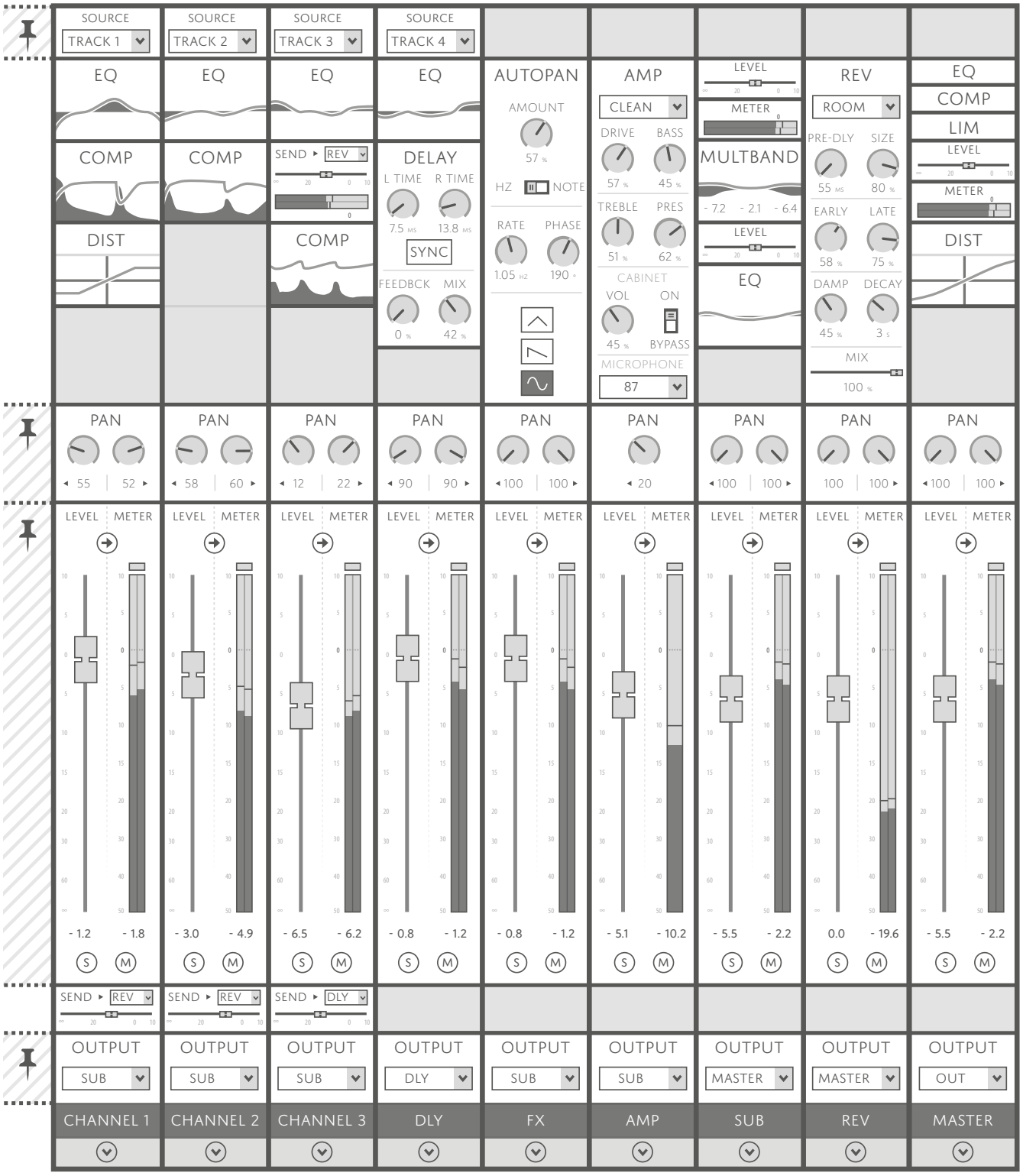


Figure 30: Routing inspector disabled. No connection lines are displayed, and therefore, the main block structure is in focus.

4.4 Interaction

The primary aim of this proposal is to offer a conceptual description of a modernised mixing interface – one that can be used with a wide array of present-day computers. The fundamental concepts of the interface have been designed to work across multiple devices, namely desktop computers and touchscreen devices. Discussion of high-level interaction is therefore mainly omitted from this proposal, and this section provides only a brief discussion of the most essential interaction methods.

The process block interface has five principal actions: (1) adding a process block, (2) removing a process block, (3) moving a process block, (4) adjusting the size of a process block, and (5) adjusting the width of a channel. The bulk of the tasks related to configuring the mixer can be handled with these five actions.

Process blocks can be added to the block column area either via a contextual menu or using a drag and drop method to "grab" the blocks from the *block browser pane*. If a block is dragged between two adjacent existing blocks, the block column is automatically rearranged. The block browser pane is illustrated in Figure 31.

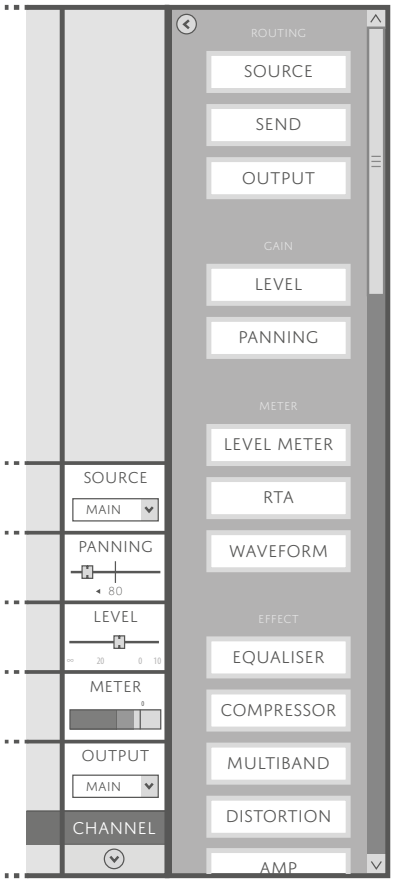


Figure 31: The block browser pane beside the main block area.

Using mouse-based interaction, clicking certain parts of the interface with the secondary mouse button shows a contextual menu, i.e. a list of possible key actions that relate to the clicked object or area. Touch-based interaction provides similar capabilities, but instead of clicking, the contextual menu can be accessed with a press and hold gesture – that is, by touching the screen with a finger and holding the finger stationary for a moment. The contextual menu for empty column area is shown in Figure 32. Alternatively, blocks can be added from the main menu[24].

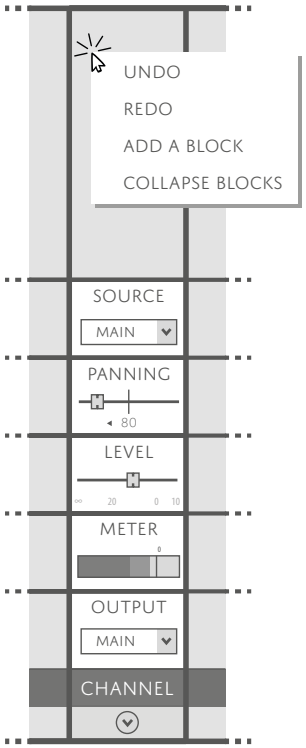


Figure 32: The contextual menu for empty block column area.

A process block can be removed via the corresponding contextual menu or by triggering a *remove block* action for the selected block from the main menu. Multiple blocks can be selected and therefore deleted simultaneously. An implementation of the process block interface should offer comprehensive reversal features for all action – that is to say, an undo action should be available at all times.

Process blocks can be moved easily with drag and drop gestures. A process block can be resized by dragging its horizontal boundaries. Similarly, the width of a channel is adjustable by dragging the side boundaries. Mouse-based interaction allows instant dragging, whereas with touchscreen devices a modal approach is necessary in order to avoid overriding the established scrolling gestures.

24 Main menus are operating system-dependent.

4.5 Addressing different devices

Several popular computing device types exist today, and the number of distinct computers may still increase in the future. For example, whereas tablet computers were novel devices in mainstream personal computing a few years ago, they are ubiquitous in everyday computing now. Today, "convertible" computers, i.e. hybrid laptop–tablet computers with hinge mechanisms or detachable keyboards, are now considered novel.

Personal computing is therefore in a state of flux. This creates an unavoidable challenge many modern software user interfaces need to address, provided the application is not meant to target only a very specific device segment.

Transferring the established audio production workflow to the computer-based system was a major ambition in the development of the original computer-based digital audio workstations. Today, the design process has to take the increasing number of distinct devices into account. Moreover, new tasks arise out of changes in music production and genres. These circumstances result in an increasing number of task–device combinations, a situation that is portrayed in Figure 33.

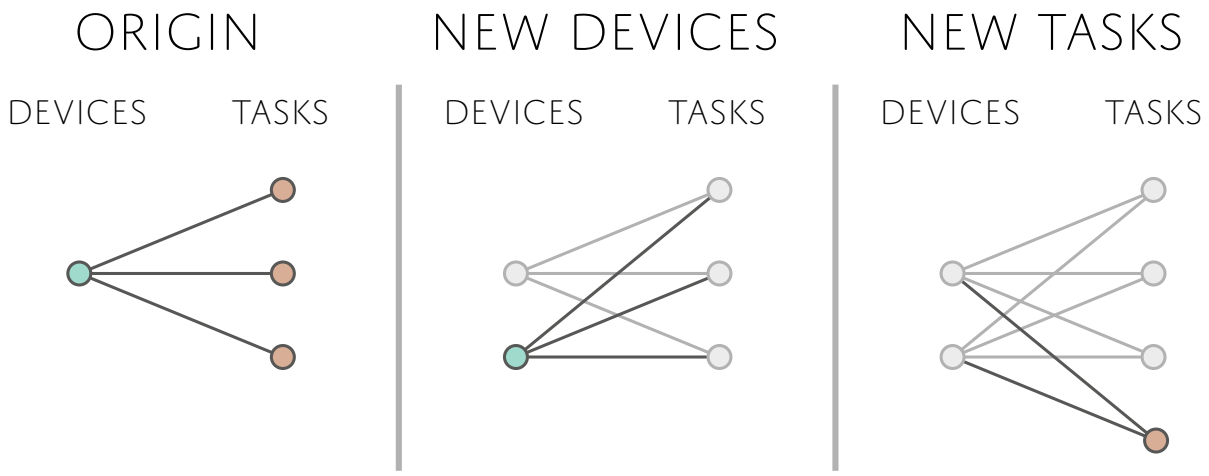


Figure 33: New device types and changes in music production result in an increasing number of task–device relations.

The growing number of task–device relations described in Figure 33 requires measures to simplify the design process and prevent the result from becoming vague. Using the relations shown in Figure 33 as a road map for an actual design would result in a very complex design process; moreover, maintenance of the software would become extremely challenging. In reality, such approach would be absurd, as different devices are used in different situations; some device types,

e.g. tablets, are often used casually, and apps that are functionally simplified may therefore be preferred.

Consequently, concrete task–device relations do not correspond with the diagram shown in Figure 33. Offering every application for every device type is not practical, let alone necessary. Instead, popular mobile apps, for example, are often lightweight versions of their traditional desktop counterparts, or offer functionality that is in close relation to mobility or touch. However, whereas designing for a specific target may be sensible at this particular moment, especially the future of the desktop computer – operated with a mouse and a keyboard – is uncertain.

As discussed in the sections 2.2 "Brief review of the technological basis" and 3.5 "Attributes of touchscreen devices", touch is now a ubiquitous input method. The traditional combination of a mouse and a keyboard is still nevertheless widely used. Additionally, the displays of both traditional desktop computers and modern touchscreen devices vary greatly in size, quality, and pixel density. Designing completely separate user interfaces for several devices is inefficient, but a single, rigid interface cannot cater for the specific needs of distinct device types. Hence, the process block structure is adaptable to changing circumstances.

The concepts presented thus far in this Chapter allow the interface to be adjusted to meet the various needs of modern-day music production. The user can choose to display extended parameters and visualisations for certain important elements, while other, non-essential elements are collapsed (see the section 4.3 "Principal features"). Alternatively, such resizing could happen automatically according to the display device; small-sized devices could minimize the blocks automatically, for instance. These features make targeting multiple different device types feasible, but furthering such support also requires the means to adjust the size of the elements without making changes to the displayed contents.

Flexible layout

Only a few years ago, the display size was a reasonable indicator of the display's pixel dimensions. However, modern display devices offer increasingly high pixel densities. There are many examples of smart phones with approximately 5" displays with pixel dimensions of 1920 × 1080 (HTC, 2013; LG Electronics, 2013; Samsung Electronics, 2013; Sony Mobile Communications, 2013). Similarly, larger

display panels, such as the ones in tablets and laptop computers, are also incorporating dense pixel grids (Apple, 2013f; Apple, 2013g; Google, 2013c).

Creating a single user interface that supports input devices of varying accuracy is challenging. As discussed in the section 3.5 "Attributes of touchscreen devices", touch-based graphical user interfaces need to take into account the size of the fingertip. Moreover, whereas mouse-based interaction has a constant-sized pointer, fingertips vary greatly in size. Touch, as an input method, inevitably results in interface design challenges, e.g. in a question of how to make sure the user is able to hit the correct target from a group of adjacent elements reliably.

The problem of touch target size has generated some research interest. Bi and Zhai (2013) proposed "Bayesian Touch Criterion", a statistical finger touch target selection criteria. Jain (2013) described a virtual fingertip library which could be used to simulate different fingertip sizes to help users overcome the limitations of the size of their own fingertips when using touchscreen devices.

Such solutions are not, however, appropriate for cases where the interface is also used with a mouse. Although touch-oriented interface elements can certainly be hit with a mouse, the interface likely includes an excess of white space or has unnecessarily large interface elements for mouse-based use; this leads to poor information density for complex applications, such as digital audio workstations.

A solution to cater for both mouse-based interaction and touchscreen interaction is to make the interface adjustable. This principle is used in, for example, Microsoft Word 2013, in which the user can choose either the *Mouse* mode with tighter interface element layout or the *Touch* mode with more space between the elements (Microsoft, 2013f).

Concepts from the field of visual arts can be used as tools for designing such interfaces. In drawing, essential compositional components include positive shapes, negative spaces, and the format (Edwards, 2008: 120). These three components are important also in graphical user interface design, where – at least for now – the interfaces must reside within the area dictated by the display device, i.e. the format.

A typical graphical user interface consists of the user interface elements, i.e. the positive shapes, and the background areas, in other words, the negative spaces. When graphical user interfaces are discussed, the positive forms often get

significant attention. However, the negative space, a concept often referred to as white space in design, is central to scalability.

The concept of negative space is used in the process block interface to support distinct computing devices. The controls and other elements inside the process blocks are enlarged, and the process blocks themselves have extra padding when the interface is used with a touch-based input device. When the interface is used with a traditional desktop computer, the layout is tighter. The touch configuration of the interface concept is shown in Figure 34, and the corresponding mouse configuration in Figure 35.

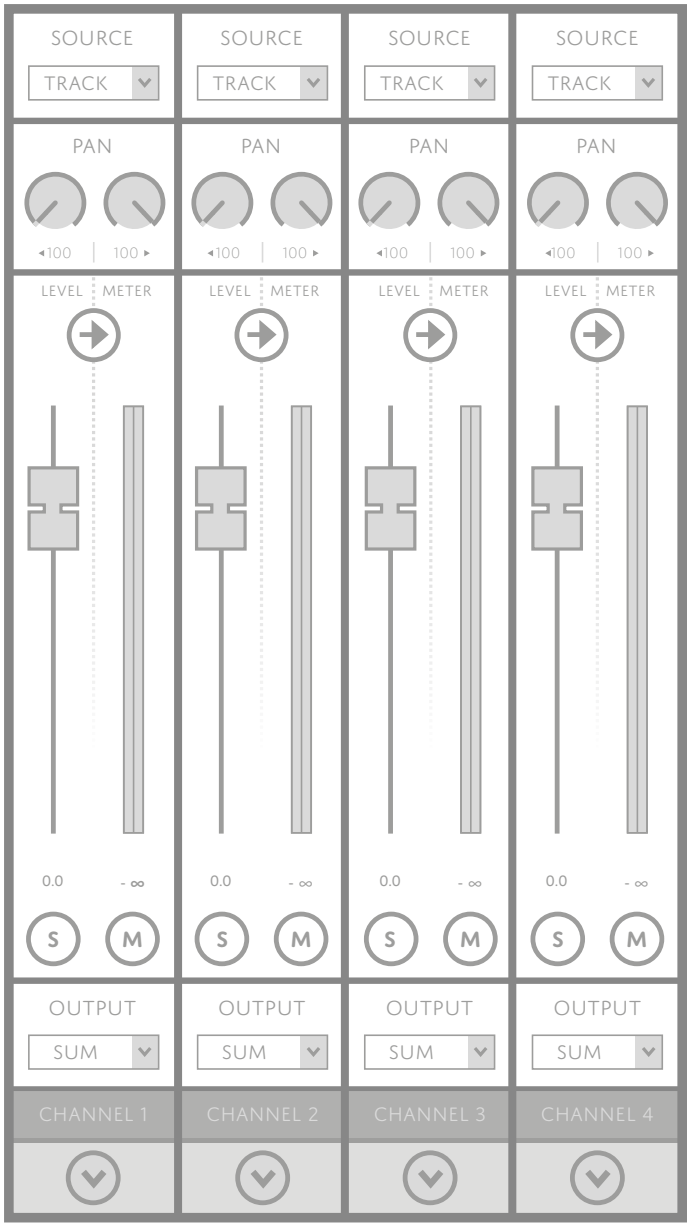


Figure 34: The process block interface in touch configuration.

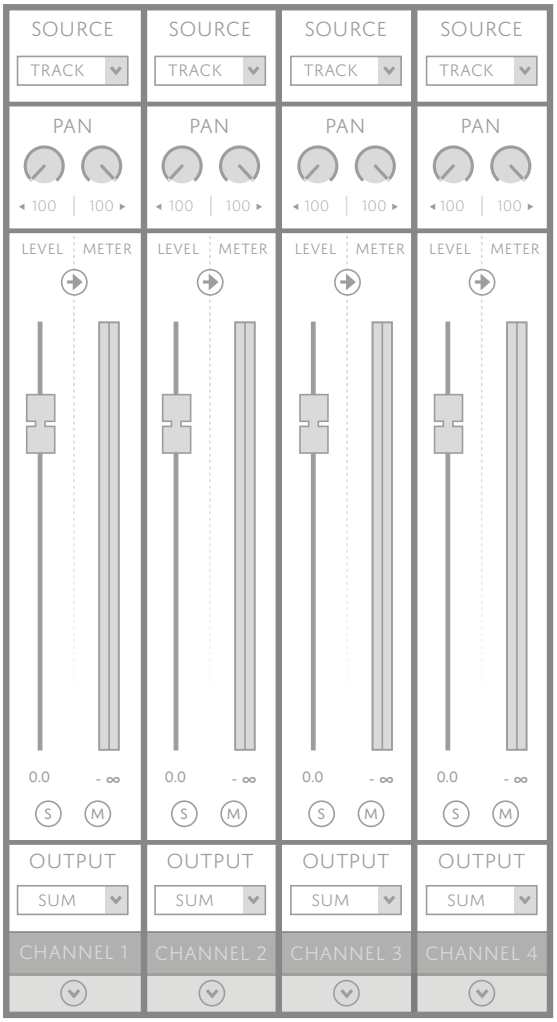


Figure 35: The process block interface in mouse configuration.

The usability of the concept can be improved further by making the interface adapt to the input device automatically. Moreover, this design is not limited to only the two configurations presented here; on the contrary, the design can be adjusted to support different user–device–task-combinations with ease.

5 Conclusions

This thesis has been concerned with whether the current, established user interface paradigms in digital audio workstations are still optimal from the standpoint of the present day. The background of these paradigms was discussed in Chapter 2 “Underlying concepts of modern audio workstations and user interfaces”, the paradigms were defined and examined in Chapter 3 “Examination of established user interface paradigms”, and methods to modernise the mixing paradigm were proposed in Chapter 4 “Processing with blocks – an interface concept for mixing”.

In this Chapter, the results of this thesis are discussed. The initial steps of the concept development are outlined in the section 5.1 “Conceptual development of the interface”. The conclusions of this thesis are presented and reflected on in the section 5.2 “Outcomes and reflection”. An outlook for the future is sketched in the section 5.3 “Future research and development”.

5.1 Conceptual development of the interface

The process of developing the interface concept presented in Chapter 4 consisted of many small steps – and a few larger leaps; in general, the design process was highly iterative. The core principles – flexible signal paths, improved visualisation, scalability, and conceptual future-proofing – remained fixed during the development of the concept, but the interface structure went through multiple renditions.

The creation of multiple distinct versions of the interface provided valuable information about the relationships between different key properties, e.g. between the clarity of the signal flow representation and the versatility of the routing system. Observing the problems that occurred when certain aspects were emphasised was vital to balance the core properties. For example, one of the initial versions of the interface (see Figure 36) offered very flexible routing possibilities, extensive zooming capabilities, a means to handle parallel processing chains within channels, and visual groupings based on the Gestalt laws of grouping. On the other hand, it risked certain central facets of usability such as learnability and efficiency.

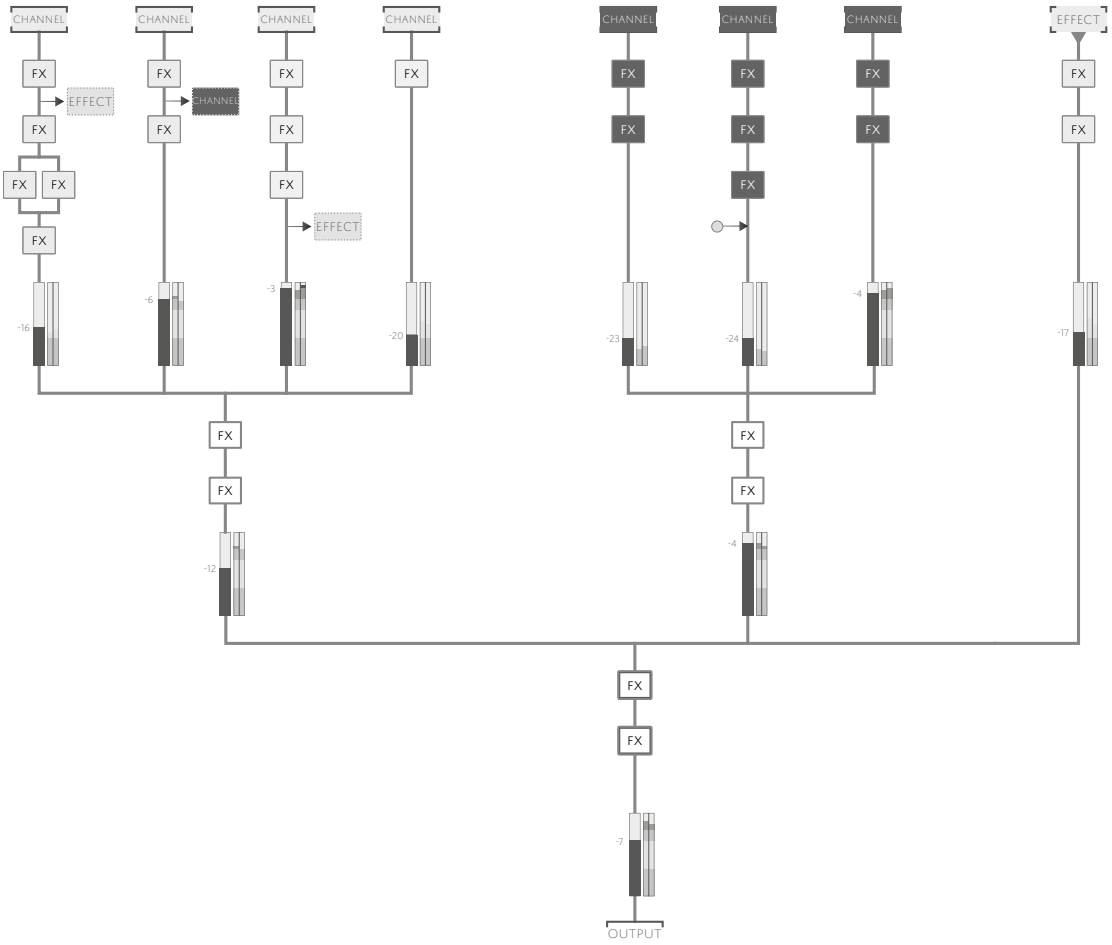


Figure 36: An early interface version. This version focused on the signal flow, groupings, and the visual representation of the summation hierarchy.

The initial conceptual development phase was based on rapid iterations, and therefore only the most prominent usability issues were considered. These design problems were then used for the formulation of the next, improved design iteration. At the beginning, concepts that would thoroughly break the convention of vertical strip-based mixing were investigated. However, these designs, e.g. the interface illustrated in Figure 36, ended up being overly complicated in terms of prospective use cases. In other words, the initial approach resulted in complex designs that, while they featured ample functionality and novel representations of the signal paths, additionally impaired basic functionality such as channel level adjustments.

A simpler approach was therefore selected for the next iterations. The flexible routing system was kept intact within individual channels, but it was combined with the established strip-based channel concept. The primary intention in these iterations was that the signal path between the channel's input and the channel's output should be evident, although the flow from a channel to another was no longer visible. This concept was prototyped with an interface based on signal path "threads" that allowed signal processing elements to be placed in any order between the source and the output (see Figure 37).

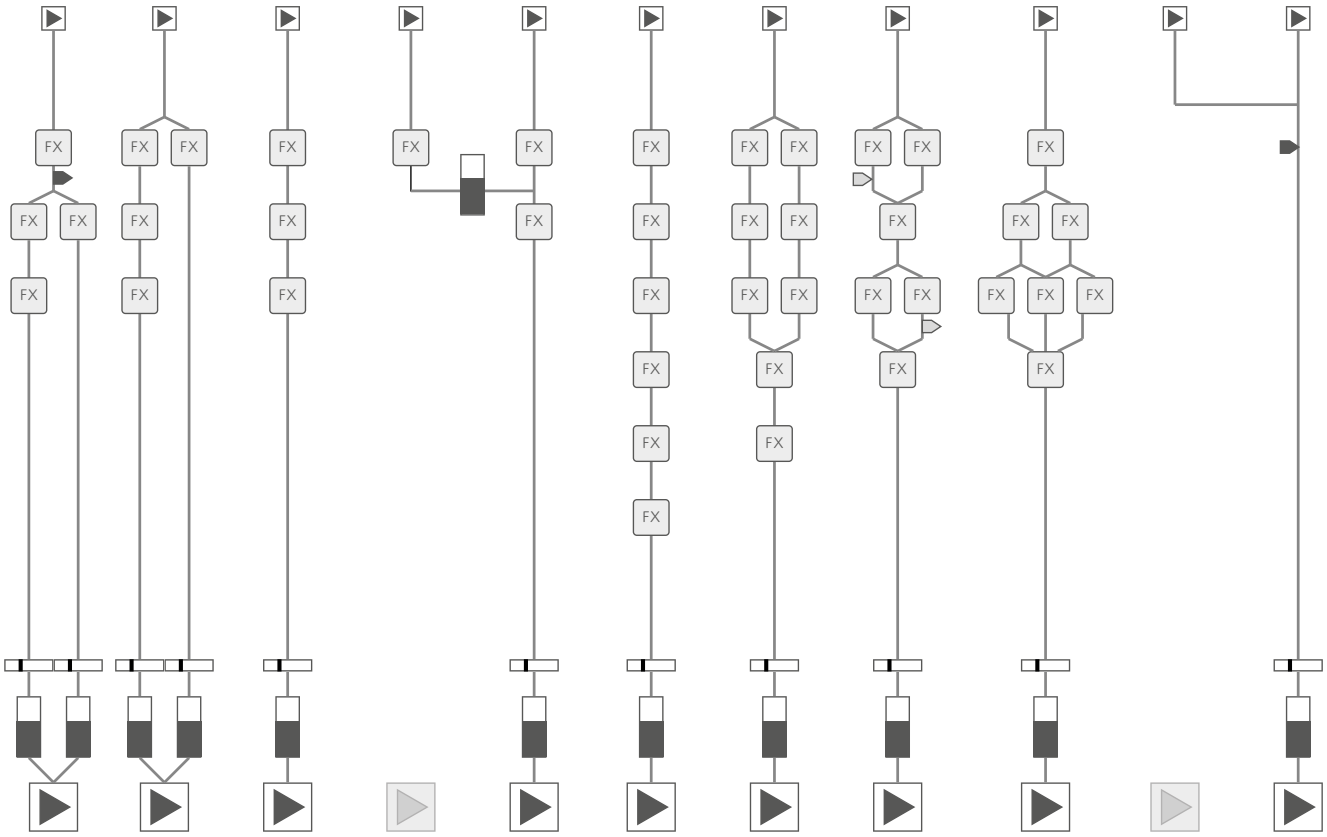


Figure 37: A simplified interface approach that combined the flexible signal paths from the initial prototype with the established concept of strip-based channel entities.

Means to represent the effects of the processing elements visually were explored in the next iterations; this capability was viewed as another marked improvement on the established mixing console paradigm. A study for the visual representations is shown in Figure 38. These interface versions demonstrated that the elements should be resizable: visualising every process permanently with a static-sized element seemed to result in excessive interface clutter and reduced effectiveness – considering the aim was to provide improved at-a-glance information.

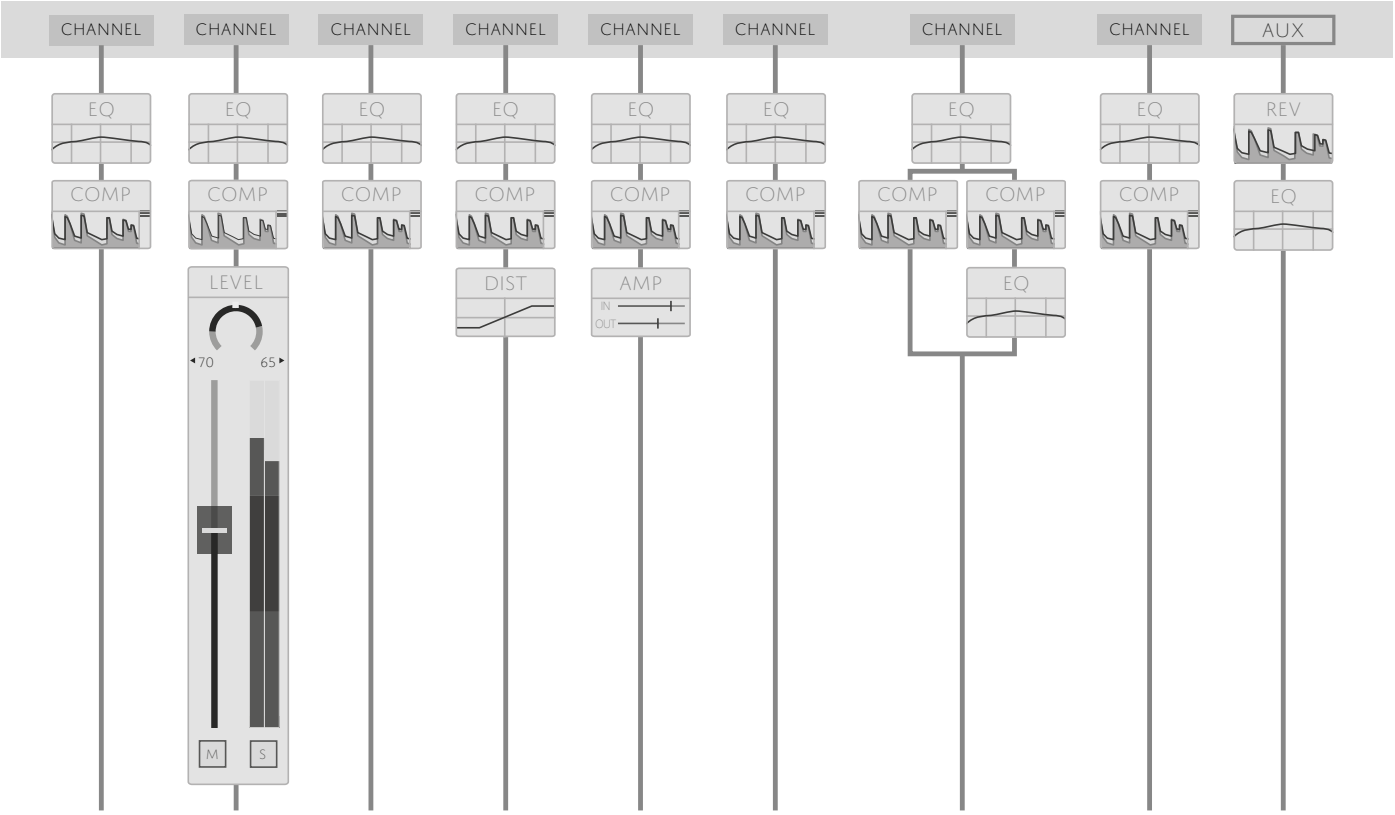


Figure 38: Visual representations of the audio processing that occurs within channels.

At this point, the design still faced a severe issue: the concept offered great flexibility of element order and size, but making use of this freedom led to disorderly layouts. The final step in the conceptual development was thus to create a structure that would support the flexible aspects of the earlier prototypes on one hand and give the interface some rigidity on the other. The result was the column-based process block structure proposed in Chapter 4, offering means to combine the newly designed flexible interface with the solid arrangement central to the established mixing console paradigm.

The interface versions discussed here revolve around vertical signal paths, but concepts utilising horizontal flow were also experimented with. In fact, the final

process block structure could be rotated to support left-to-right paths – or right-to-left, for that matter – instead of the top-to-bottom flow. The decision to select the vertical flow was based on conventions: if necessary, the block interface can simulate the traditional console paradigm quite thoroughly, which provides familiarity.

5.2 Outcomes and reflection

Vast technological changes have taken place since the initial computer-based digital audio workstation systems were developed, yet some of the most fundamental interfacing paradigms have remained mostly unaltered. A thorough examination of these interface paradigms, including their background, formed an essential part of this thesis. This examination provided a solid foundation for the other main topic of this thesis: the development of the modernised, block-based user interface concept for mixing in digital audio workstations.

Inspecting the user interfaces of multiple prominent digital audio workstations was central to the formulation of the paradigm abstractions, and studying relevant previous research and looking at modern technology closely formed a framework by which to evaluate the abstractions. The track-oriented timeline paradigm and the mixing console-based paradigm, two distinct structural interface paradigms, were identified in the examination presented in Chapter 3 "Examination of established user interface paradigms". The findings correspond to what has been found in a previous study: Duignan (2008: 59) stated that Digidesign[25] Pro Tools, Apple Logic, and Steinberg Cubase are all based on the "ubiquitous multitrack-mixer metaphor".

The basis for the established paradigms was inspected in Chapter 2 "Underlying concepts of modern audio workstations and user interfaces", and three distinct hardware precursors of the digital audio workstations were identified: the sequencer, the multitrack recorder, and the analogue mixing console. As discussed in Chapter 2, this background demystifies the current interface structure: the development of the digital audio workstations and the specialised audio production devices were closely intertwined.

25 Avid acquired Digidesign already in 1995 (Avid Technology, 2013a), but the brand was renamed later.

The process block-based interface concept, discussed in Chapter 4 "Processing with blocks – an interface concept for mixing", was proposed as a means to modernise the traditional console-based mixing interface. The term *process* was a cornerstone of the interface proposal. This term was used in the concept to refer to an abstract mixing operation, e.g. using an effect plug-in, sending a signal from one channel to another, and adjusting the level of a channel.

The interface concept incorporated features unavailable in the established paradigms, for example, the possibility to arrange the signal chain elements without restraints. The result was an interface structure that allows the user to make mixing-related decisions more freely in comparison to traditional digital audio workstations, and in addition, the block-based interface concept supports personal computing devices other than desktop computers.

The traditional, rigid mixing console-based structure is therefore not the only way to approach the mixing paradigm in a digital audio workstation, as the process block concept demonstrated. An interface that is both flexible and compatible with the established music production procedures is achievable by discarding some of the most prominent currently used interface metaphors.

Level metering was in this text considered a significant paradigm in itself, although meter elements are typically included in the fundamental structural paradigms. Current meter designs proved to be somewhat conflicting: traditional interface elements are used for representing a dynamic range which, in current digital audio workstation systems, is immensely large. Moreover, from the perspective of user interface design, metering is governed by the rigid interface structure, which poses restrictions on the size and placement of the meter elements.

A heuristic approach was applied to the user interface examination, which inevitably introduced a certain amount of subjectiveness. Ascribable to the author's background, this thesis contained an implicit viewpoint: that of an active user of the digital audio workstation systems. Evaluating the success of the interface concept is nevertheless challenging, as although specific advantages over the traditional paradigms were demonstrated, usability testing involving a number of users from the targeted user group would be required to draw any definite conclusions. The recent technological advancements give nonetheless confidence that the interface paradigms in digital audio workstations need to be modernised, as this thesis proposed.

5.3 Future research and development

The block-based interface structure was presented as a conceptual prototype, and although the discussion leaned on the possibilities offered by the present-day devices, the aim was to frame an abstract version of the interface concept. The implicit idea in the block-based structure was to provide for new forms of human–computer interaction in addition to current desktop computers and touch-based devices. In the future, supporting a wide array of different input and output methods might be an integral part of user interface design. Norman and Wadia (2013: sec. 2, para. 1) have discussed such prospects of interaction design:

> "Today, we talk of the "interface" between people and products with the assumption that it is a physical presence, a panel or otherwise visible structure with which people interact. In fact, calling something a "touch" or "multi-touch" interface implies a physical structure that is intended to be touched. However, as we move forward, the options expand beyond mere physical touch. We might allow interaction at any location on a device, or even without touching. The new design considerations must apply to interface inputs that use touch or not (touchless) with outputs that can involve any medium or sensory modality."

On a concrete level, some design decisions were made on the basis of assertions. For example, a vertical channel structure was used instead of horizontal channels to offer compatibility with the established mixing console paradigm. The assertion was that the familiarity achieved with this decision improves the usability of the interface concept, but this approach could be challenged. Weighing the vertical channel structure against a horizontal one methodologically would presumably provide useful information about the differences between the two, and could open up significant possibilities to refresh the established paradigms yet more thoroughly. The current track-based timeline views are based on horizontal tracks; using a horizontal channel structure in the mixing view would enable linking the two views. Inspecting this topic could even lead to insights on whether two separate views are necessary.

Different methods could be used for the further development of the interface concept. As the interface prototype proposed in this thesis was fundamentally conceptual, creation of concrete and operable renditions seems a logical next step. However, the degree of fidelity used for the next interface prototypes should be considered, as it likely affects the nature of the process considerably.

Developing a high-fidelity prototype – that is to say, a rendition that would represent the actual interface, including its interaction-related aspects – would take significantly more time and effort in comparison to low-fidelity methods. Low-fidelity prototyping, e.g. paper prototypes, would likely provide useful information about the usability of the proposed structure, and this information would be available promptly. The use of low-fidelity prototyping would therefore allow making changes to the interface iteratively, as the prototyping cycle could be kept short. However, development of a high-fidelity version should follow this phase; this is essential for evaluating the aspects of the interaction design.

A usability testing session – or possibly multiple such sessions – should be carried out in any case. In the field of usability research, the distinction between the designer and the user is often emphasised: the feedback from the target audience should not be belittled. Even the deepest insights of a single person are incapable of replacing the information provided by a group of target users. Therefore, a well-documented testing session carried out by a group of evaluators would be essential for further development of the interface concept.

The process block structure was designed to be a highly expandable environment in general, but one concept above all was intended for extensive future development: the effect block. The process block interface utilises highly scalable design, and the ability to zoom in could be intensified so that the actual contents of effect blocks become visible. Proprietary effect blocks consisting of individual components could be included, and these components could be used for creating a wide array of different blocks. In other words, the block structure could act as a modular system with the ability to encapsulate specific combinations of components in process blocks.

The track-based timeline view was recognised as a seminal interface paradigm in this thesis. However, the track view was not part of the interface concept, as the inclusion of a subject of such magnitude would have resulted in a lengthy discussion – which would have gone beyond the scope of this thesis. The track environment has nevertheless remained quite static, and therefore, refreshing the track-based timeline paradigm could provide fruitful results.

The process block-based mixing interface proposed in this thesis was in fact designed to support a redesigned source view – that is to say, a modernised track view. This is one of the primary reason why the traditional channel input field was

replaced with the more flexible source block. A modern track view could offer, for example, means to tag regions, and the tags could be used for linking the regions to specific channels in the block view. Therefore, although a source can be a traditional track, it can also be something drastically different.

References

Ableton, 2013. *Live 9 Suite*. (9.0.4) [computer software] Available at: <https://www.ableton.com/en/shop> [Accessed 27 June 2013].

Apple, 2008. *Logic Pro*. (8.0.2) [computer software] Apple.

Apple, 2010. *Apple Launches iPad*. [press release] 27 January 2010. Available at: <http://www.apple.com/pr/library/2010/01/27Apple-Launches-iPad.html> [Accessed 13 June 2013].

Apple, 2013a. *iOS 7*. [video online] Available at: <http://www.apple.com/ios/design> [Accessed 17 December 2013].

Apple, 2013b. *OS X Mavericks*. [online] Available at: <http://www.apple.com/osx/preview> [Accessed 13 June 2013].

Apple, 2013c. *Logic Pro X: user guide*. [pdf] Available at: <http://manuals.info.apple.com/MANUALS/1000/MA1648/en_US/logic_pro_x_user_guide.pdf> [Accessed 16 September 2013].

Apple, 2013d. *iOS 7: what's new*. [online] Available at: <http://www.apple.com/ios/whats-new> [Accessed 17 December 2013].

Apple, 2013e. *GarageBand*. [online] Available at: <http://www.apple.com/apps/garageband> [Accessed 24 September 2013].

Apple, 2013f. *iPad*. [online] Available at: <http://www.apple.com/ipad/specs> [Accessed 21 October 2013].

Apple, 2013g. *MacBook Pro*. [online] Available at: <http://www.apple.com/macbook-pro/specs-retina> [Accessed 21 October 2013].

Apple, 2014. *Logic Remote*. [online] Available at: <https://itunes.apple.com/us/app/logic-remote/id638394624?mt=8> [Accessed 24 February 2014].

Avid Technology, 2011a. *Pro Tools* (9.0.6) [computer software] Avid Technology.

Avid Technology, 2011b. *Pro Tools|HDX*. [datasheet] Available at: <https://www.avid.com/static/resources/common/documents/datasheets/pro_tools_HDX_ds_US_sec_2012_03_06.pdf> [Accessed 20 December 2013].

Avid Technology, 2013a. *Corporate Profile*. [online] Available at: <http://www.avid.com/US/about-avid/corporate-profile> [Accessed 28 January 2014].

Avid Technology, 2013b. *Pro Tools Family*. [online] Available at: <http://www.avid.com/US/products/family/Pro-Tools> [Accessed 13 June 2013].

Avid Technology, 2013c. *Pro Tools* (11.0.2) [computer software] Available at: <http://shop.avid.com/store/product.do?product=325865714796064> [Accessed 9 November 2013].

Avid Technology, 2013d. *Avid S6: mixing redefined*. [pdf] Available at: <http://www.avid.com/static/resources/common/documents/datasheets/S6_ds_A4.pdf> [Accessed 20 October 2013].

Avid Technology, 2013e. *Pro Tools 11: advanced metering*. [video online] Available at: <http://www.avid.com/US/products/pro-tools-software> [Accessed 25 September 2013].

Avid Technology, 2013f. *What's new in Pro Tools and Pro Tools HD: version 11.0*. [pdf] Available at: <http://avid.force.com/pkb/articles/en_US/download/Pro-Tools-11-Whats-New-Guide> [Accessed 10 September 2013].

BBC, 2007. How the CD was developed. *BBC NEWS*, [online] Last updated: 17 August 2007. Available at: <http://news.bbc.co.uk/2/hi/technology/6950933.stm> [Accessed 8 January 2014].

Bi, X. and Zhai, S., 2013. Bayesian touch: a statistical criterion of target selection with finger touch. In: *Proceedings of the 26th annual ACM symposium on user interface software and technology (UIST '13)*, pp. 51–60. St. Andrews, United Kingdom 8–11 October 2013. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 22 October 2013].

Bitwig, 2013a. *Bitwig Studio*. [online] Available at: <https://www.bitwig.com/en/bitwig-studio> [Accessed 20 December 2013].

Bitwig, 2013b. *Bounce and slice in Bitwig Studio*. [video online] Available at: <http://www.youtube.com/watch?v=Dy-8u6EAt-I> [Accessed 20 December 2013].

Blackwell, A. F., 2006. The reification of metaphor as a design tool. *ACM Transactions on Computer-Human Interaction (TOCHI)*, [e-journal] 13(4), pp. 490–530. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 23 August 2013].

Blip Interactive, 2010. *NanoStudio: a virtual recording studio for iOS*. [online] Available at: <http://www.blipinteractive.co.uk/index.php> [Accessed 24 September 2013].

Buchholz, W., 1962. Choosing a number base. In: W. Buchholz, ed. 1962. *Planning a computer system: project stretch*. [pdf] New York: McGraw-Hill Book Company, pp.42–59. Available at: <http://amturing.acm.org/Buchholz_102636426.pdf> [Accessed 28 June 2013].

Cabral, M. C., Morimoto, C. H., and Zuffo, M. K., 2005. On the usability of gesture interfaces in virtual reality environments. In: *Proceedings of the 2005 Latin American conference on Human-computer interaction (CLIHC '05)*, pp. 100–108. Cuernavaca, México 23–26 October 2005. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 29 July 2013].

Cakewalk, 2013a. *Cakewalk DAW labs: the best computers for recording and editing digital audio*. [online] Available at: <http://www.cakewalk.com/PCResource> [Accessed 13 June 2013].

Cakewalk, 2013b. *Sonar X3: perform* [online] Available at: <http://www.cakewalk.com/products/sonar/features.aspx?v=perform> [Accessed 20 October 2013].

Cakewalk, 2013c. *Sonar X3: mix* [online] Available at: <http://www.cakewalk.com/products/sonar/features.aspx?v=mix> [Accessed 20 October 2013].

Cakewalk, 2013d. *SONAR core technology and 64 bit double precision engine*. [online] Available at: <http://www.cakewalk.com/Products/feature.aspx/SONAR-Core-Technology-and-64-bit-Double-Precision-Engine> [Accessed 20 December 2013].

Cakewalk, 2013e. *Sonar X3: experience touch*. [online] Available at: <https://www.cakewalk.com/products/sonar/touch.aspx> [Accessed 21 December 2013].

Carlos, M. and Stewart, T., 1983. *Fairlight CMI Page R Real Time Composer: operation manual*. [manual] Sydney: Fairlight Instruments.

Celemony Software, 2013a. *About Celemony*. [online] Available at: <http://www.celemony.com/cms/index.php?id=1000> [Accessed 30 July 2013].

Celemony Software, 2013b. *Five years of DNA: a retrospective*. [online] Available at: <http://www.celemony.com/cms/index.php?id=dna_anniversary> [Accessed 23 August 2013].

Chen, F., Koufaty, D. A., and Zhang, X., 2009. Understanding intrinsic characteristics and system implications of flash memory based solid state drives. In: *Proceedings of the eleventh international joint conference on measurement and modeling of computer systems (SIGMETRICS '09)*, pp. 181–192. Seattle, USA 15–19 June 2009. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 2 August 2013].

Chen, P. M., Lee, E. K., Gibson, G. A., Katz, R. H., and Patterson, D. A., 1994. RAID: high-performance, reliable secondary storage. *ACM Computing Surveys*, [e-journal] 26(2), pp. 145–185. Available through: ACM Digital Library <dl.acm.org> [Accessed 2 August 2013].

Cockos, 2013. *Reaper*. (4.581) [computer software]. Available at: <http://www.reaper.fm/purchase.php> [Accessed 21 December 2013].

Cycling '74, 2013a. *Max*. (6.1.1) [computer software] Available at: Cycling '74 Shop <http://cycling74.com/shop> [Accessed 16 September 2013].

Cycling '74, 2013b. *Max*. [online] Available at: <http://cycling74.com/products/max/> [Accessed 16 September 2013].

David, P. A., 1985. Clio and economics of QWERTY. *The American economic review: papers and proceedings of the ninety-seventh annual meeting of the American economic association*, [e-journal] 75(2), pp. 332–337. Available through: JSTOR <http://www.jstor.org/stable/1805621> [Accessed 26 June 2013].

Detune, 2013. *Mo1D*. [online] Available at: <http://www.detune.co.jp/korgmo1d.html> [Accessed 24 July 2013].

Duignan, M., 2008. *Computer mediated music production: a study of abstraction and activity*. PhD thesis, Victoria University of Wellington. [pdf] Available at: <http://researcharchive.vuw.ac.nz/handle/10063/590> [Accessed 16 August 2013].

Duignan, M., Noble, J., Barr, P., and Biddle, R., 2004. Metaphors for electronic music production in Reason and Live. In: M. Masoodian, S. Jones and B. Rogers, eds., *Proceedings of 6th Asian Pacific Conference on Computer Human Interaction (APCHI '04)*, pp. 111–120. Rotorua, New Zealand 29 June – 2 July 2004. Berlin: Springer-Verlag.

Duignan, M., Noble, J., and Biddle, R., 2005. A taxonomy of sequencer user-interfaces. In: ICMA (International Computer Music Association), *International Computer Music Conference Proceedings*. Barcelona, Spain 5–9 September 2005. [pdf] Available at: <http://quod.lib.umich.edu/i/icmc/bbp2372.2005?rgn=full+text> [Accessed 16 August 2013].

Edwards, B., 2008. *The new drawing on the right side of the brain*. London: HarperCollins. (original work published 1979)

Electronic Music Studios, 197?. *Electronic Music Studios: the SYNTHI Sequencer 256*. [manual] London: Electronic Music Studios.

FabFilter, 2013. *Pro-C*. (1.23) [computer software] FabFilter. Available at: <http://www.fabfilter.com/download/pro-c-compressor-plug-in> [Accessed 17 November 2013].

Farhadi-Niaki, F., GhasemAghaei, R., and Arya, A., 2012. Empirical study of a vision-based depth-sensitive human-computer interaction system. In: *Proceedings of the 10th Asia Pacific conference on computer human interaction (APCHI '12)*, pp. 101–108. Matsue, Japan 28–31 August 2012. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 29 July 2013].

Fischer, R., 2013. *TouchOSC*. (1.9.3) [mobile application] hexler. Available at: <https://itunes.apple.com/app/touchosc/id288120394?mt=8> [Accessed 24 February 2014].

Gartner, 2013. *Gartner says worldwide PC, tablet and mobile phone combined shipments to reach 2.4 billion units in 2013*. [press release] 4 April 2013. Available at: <http://www.gartner.com/newsroom/id/2408515> [Accessed 13 June 2013].

Gavin, R., 2014. OneDrive for Everything in Your Life. *The OneDrive Blog*, [blog] 27 January. Available at: <http://blog.onedrive.com/onedrive-for-everything-your-life> [Accessed 4 February 2014].

Gibson, J. J., 1986. *The ecological approach to visual perception*. New York: Psychology Press.

Gohlke, K., Hlatky, M., Heise, S., Black, D., and Loviscach, J., 2010. Track displays in DAW software: beyond waveform views. In: Audio Engineering Society, *128th Convention*. London, United Kingdom 22–25 May 2010. [pdf] Available through: AES E-Library <http://www.aes.org/e-lib/> [Accessed 14 August 2013]

Goldberg, D., 1991. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys (CSUR)*, [e-journal] 23(1), pp. 5–48. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 9 September 2013].

Google, 2013a. *Android*. (4.4.2) [operating system].

Google, 2013b. *Android*. (4.2.2) [operating system].

Google, 2013c. *Chromebook Pixel*. [online] Available at: <http://www.google.com/intl/fi/chrome/devices/chromebook-pixel> [Accessed 21 October 2013].

Halasz, F. and Moran, T. P., 1982. Analogy considered harmful. In: *Proceedings of the 1982 conference on human factors in computing systems (CHI '82)*, pp. 383–386. Gaithersburg, Maryland, United States 1982. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 13 August 2013].

Henke, R., 201?. *Ableton Live*. [online] Available at: <http://www.monolake.de/technology/ableton_live.html> [Accessed 12 January 2014].

Holmes, G., 1997. *Page R*. [image online] Available at: <http://www.ghservices.com/gregh/fairligh/page_r.gif> [Accessed 20 January 2014].

Holmes, G., 2010. *The Fairlight CMI*. [online] (Created: 1 October 1996; Last updated: 18 September 2010) Available at: <http://www.ghservices.com/gregh/fairligh/#seriesIIsoftware> GH Services. [Accessed 15 January 2014].

HTC, 2013. *HTC One: specs*. [online] Available at: <http://www.htc.com/www/smartphones/htc-one/#specs> [Accessed 21 October 2013].

Huber, D. M. and Runstein, R. E., 2005. *Modern recording techniques*. 6th ed. Amsterdam: Focal Press.

IEEE Computer Society, 2008. *IEEE Std 754-2008 IEEE standard for floating-point arithmetic*. The Institute of Electrical and Electronics Engineers [online] Available through: IEEE Xplore Digital Library <http://ieeexplore.ieee.org> [Accessed 9 September 2013].

Image-Line Software, 2013a. *FL Studio history*. [online] Available at: <http://www.image-line.com/blog/fl-history.php> [Accessed 28 June 2013].

Image-Line Software, 2013b. *FL Studio Mobile*. [online] Available at: <http://www.image-line.com/documents/flstudiomobile.html> [Accessed 24 July 2013].

Jain, A., 2013. Touch target optimization technique using virtual finger-tip library. *ACM SIGSOFT Software Engineering Notes*, [e-journal] 38(3), pp. 1–9. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 22 October 2013].

Jeffries, R., Miller, J. R., Wharton, C., and Uyeda, K. M., 1991. User interface evaluation in the real world: a comparison of four techniques. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '91)*, pp. 119–124. New Orleans, United States 27 April–02 May 1991. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 16 November 2013].

Johnston, W. M., Hanna, J. R. P., and Millar, R. J., 2004. Advances in dataflow programming languages. *ACM Computing Surveys (CSUR)*, [e-journal] 36(1), pp. 1–34. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 16 September 2013].

Jussila, M. and Finnvox Studiot, 2013. *Solid State Logic 4040 G*. [photograph] (Mika Jussila, Finnvox Studiot)

Khoury, G. R. and Simoff, S. J., 2003. Elastic metaphors: expanding the philosophy of interface design. In: J. Weckert and Y. Al-Saggaf, eds., *Selected papers from conference on computers and philosophy (CRPIT '03)*, pp. 65–71. Darlinghurst, Australia: Australian Computer Society. Available through: ACM Digital Library <dl.acm.org> [Accessed 19 August 2013].

Kim, S-C., Israr, A., and Poupyrev, I., 2013. Tactile rendering of 3D features on touch surfaces. In: *Proceedings of the 26th annual ACM symposium on user interface software and technology (UIST '13)*, pp. 531–538. St. Andrews, United Kingdom 8–11 October 2013. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 17 October 2013].

Korg, 2013. *iMS20*. [online] Available at: <http://www.korg.com/ims20> [Accessed 24 July 2013].

Lavry Engineering, 2012. *AD122-96 MKIII: tech specs*. [online] Available at: <http://www.lavryengineering.com/products/pro-audio/ad122-96-mkiii.html> [Accessed 10 September 2013].

Leap Motion, 2013. *Leap Motion*. [online] Available at: <https://www.leapmotion.com> [Accessed 13 June 2013].

LG Electronics, 2013. *G2: specifications*. [online] Available at: <http://www.lg.com/global/g2/sub3.html> [Accessed 21 October 2013].

Manning, P., 1993. *Electronic and computer music*. 2nd ed. Oxford: Clarendon Press.

Maschmeyer, R. and Cameron, G., Apple Inc., 2012. *Methods and systems for providing haptic control*. U.S. Patent Application 20120105333.

Metric Halo, 2013. *MIO Console*. (5.6.01) [computer software] Available at: <http://mhsecure.com/metric_halo/support/downloads.html> [Accessed 16 September 2013].

Microsoft, 2013a. *Introducing Xbox One*. [online] Available at: <http://www.xbox.com/en-US/xboxone/meet-xbox-one> [Accessed 13 June 2013].

Microsoft, 2013b. *Windows Phone*. (8.0) [operating system].

Microsoft, 2013c. *Windows*. [online] Available at: <http://windows.microsoft.com/en-us/windows/home> [Accessed 13 June 2013].

Microsoft, 2013d. *Features of Windows 8*. [online] Available at: <http://windows.microsoft.com/en-us/windows-8/features> [Accessed 8 July 2013].

Microsoft, 2013e. *Windows 8*. (8.1) [operating system] Available at: <http://windows.microsoft.com/en-us/windows/buy> [Accessed 17 December 2013].

Microsoft, 2013f. *Word 2013*. [computer software] Available at: <http://office.microsoft.com/en-001/store/?CTT=97> [Accessed 23 November 2013].

MIDI Manufacturers Association, 200?. *The complete MIDI 1.0 detailed specification*. [online] Available at: <http://www.midi.org/techspecs/midispec.php> [Accessed 1 October 2013].

Moore, F. R., 1988. The dysfunctions of MIDI. *Computer Music Journal*, [e-journal] 12(1), pp. 19–28. Available through: JSTOR <http://www.jstor.org> [Accessed 15 August 2013].

Morris, P., 1999. *Nonlinear editing: media manual*. Oxford: Focal Press.

MOTU, 2011. *Digital Performer*. (7.24) [computer software] MOTU.

MOTU, 2013. *Digital Performer*. (8.05 trial version) [computer software] Available at: <http://www.motu.com/download> [Accessed 13 December 2013].

MOTU, 2014. *Features*. [online] Available at: <http://www.motu.com/products/software/dp/features.html> [Accessed 24 February 2014].

Nash, C., 2011. *Supporting virtuosity and flow in computer music*. PhD Dissertation, University of Cambridge. [pdf] Available at: <http://revisit.info/files/PhD_Thesis_600dpi.pdf> [Accessed 12 August 2013].

Nielsen, J., 1993. *Usability engineering*. San Francisco: Academic Press.

Nielsen, J., 2012. *Windows 8: disappointing usability for both novice and power user*. [online] Available at: <http://www.nngroup.com/articles/windows-8-disappointing-usability> [Accessed 30 September 2013].

Nielsen, J. and Molich, R., 1990. Heuristic evaluation of user interfaces. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*, pp. 249–256. Seattle, United States 1–5 April 1990. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 10 November 2013].

Norman, D. A., 1998. *The design of everyday things*. London: The MIT Press. (original work published 1988)

Norman, D. and Wadia, B., 2013. *Opportunities and challenges for touch and gesture-based systems*. [online] Available at: <http://www.jnd.org/dn.mss/opportunities_and_ch.html> [Accessed 17 September 2013].

Oppenheim, A. V. and Schafer, R. W., 1975. *Digital signal processing*. London: Prentice-Hall International.

Oxford University Press, 2005. *Oxford dictionary of English*. 2nd ed., revised. Oxford: Oxford University Press.

Parhi, P., Karlson, A. K., and Bederson, B. B., 2006. Target size study for one-handed thumb use on small touchscreen devices. In: *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services (MobileHCI '06)*, pp. 203–210. Espoo, Finland 12–15 September 2006. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 24 September 2013].

Park, Y. S., Han, S. H., Park, J., and Cho, Y., 2008. Touch key design for target selection on a mobile phone. In: G. H. ter Hofte, I. Mulder and B. E. R. de Ruyter, eds., *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services (MobileHCI '08)*, pp. 423–426. Amsterdam, The Netherlands 2–5 September 2008. New York: ACM. Available through: ACM Digital Library <dl.acm.org> [Accessed 24 September 2013].

Patterson, D. A. and Hennessy, J. L., 1996. *Computer architecture: a quantitative approach*. 2nd ed. San Francisco: Morgan Kaufmann.

Pd community, 2013. *Pd*. (0.44.0) [computer software] Available at: <http://puredata.info/downloads/pure-data> [Accessed 16 September 2013].

Plogue Art et Technologie, 2013. *Bidule Standalone*. (0.9733) [computer software] Available at: <http://www.plogue.com/downloads> [Accessed 16 September 2013].

Preece, J., Rogers, Y., and Sharp, H., 2002. *Interaction design: beyond human–computer interaction*. New York: John Wiley & Sons.

PreSonus Audio Electronics, 2013. *FireStudio Project: tech specs*. [online] Available at: <http://www.presonus.com/products/FireStudio-Project/techspecs> [Accessed 10 September 2013].

Propellerhead Software, 2013. *Creative flow*. [online] Available at: <http://www.propellerheads.se/products/reason/index.cfm?fuseaction=get_article&article=features_creative_flow> [Accessed 28 June 2013].

Raskin, J., 2000. *The humane interface: new directions for designing interactive systems*. Boston: Addison-Wesley.

Robjohns, H., 1997. Anatomy of a mixer: exploration. *Sound on Sound*, [online] April 1997. Available at: <http://www.soundonsound.com/sos/1997_articles/apr97/mixeranatomy.html> [Accessed 1 October 2013].

Robjohns, H., 2000. Interfacing analogue & digital equipment. *Sound on Sound*, [online] May 2000. Available at: <http://www.soundonsound.com/sos/may00/articles/digital.htm> [Accessed 20 December 2013].

Robjohns, H., 2010. Digidesign Pro Tools. In: The SOS Team. 25 products that changed recording. *Sound On Sound*, [online] November 2010. Available at: <http://www.soundonsound.com/sos/nov10/articles/25-milestone-products.htm> [Accessed 13 June 2013].

Rock, I. and Palmer, S., 1990. The legacy of gestalt psychology. *Scientific American*, [pdf] 263(6), pp. 84–90. Available at: <http://music.dartmouth.edu/~larry/UCSC_CLASSES_2011_2012/Music_150x/readings/week_6_temporal_gestalts/rock_palmer_gestalt_psychology.pdf> [Accessed 13 August 2013].

Rossing, T. D., Moore, F. R., and Wheeler, P. A., 2002. *The science of sound*. 3rd ed. San Francisco: Addison Wesley.

Samsung Electronics, 2013. *Samsung GALAXY S4: specifications*. [online] Available at: <http://www.samsung.com/global/microsite/galaxys4> [Accessed 21 October 2013].

Sensomusic, 2013. *Usine Hollyhock*. [online] Available at: <http://www.sensomusic.com/usine> [Accessed 17 September 2013].

Shannon, C. E., 1948. A mathematical theory of communication. Reprint from: *The Bell System technical journal*, vol. 27, pp. 379–423, 623–656. [pdf] Available at: <http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf> [Accessed 8 July 2013].

Shapiro, C. and Varian, H. R., 1999. *Information rules: a strategic guide to the network economy*. Boston: Harvard Business School Press.

Shneiderman, B., 1998. *Designing the user interface: strategies for effective human-computer interaction*. 3rd ed. Reading, MA: Addison-Wesley.

Sinkkonen, I., Kuoppala, H., Parkkinen, J., and Vastamäki, R., 2006. *Käytettävyyden psykologia*. 3rd ed., revised. Helsinki: Edita.

Slate Pro Audio, 2013a. *RAVEN MTX*. [online] Available at: <http://www.slateproaudio.com/products/raven-mtx> [Accessed 21 December 2013].

Slate Pro Audio, 2013b. *RAVEN MTi*. [online] Available at: <http://www.slateproaudio.com/products/raven-mti> [Accessed 21 December 2013].

Smith, J. O., 2007. *Mathematics of the discrete Fourier transform (DFT) with audio applications*. 2nd ed. [online book] Available at: <http://ccrma.stanford.edu/~jos/mdft> [Accessed 20 December 2013].

Solid State Logic, 1988. *G series master studio system: console operator's manual*. [manual] Oxford: Solid State Logic.

Solid State Logic, 2013a. *Duality SE*. [online] Available at: <http://www.solid-state-logic.com/music/duality/> [Accessed 25 July 2013].

Solid State Logic, 2013b. *AWS 924 & AWS 948*. [online] Available at: < http://www.solid-state-logic.com/music/aws> [Accessed 25 July 2013].

Sony Mobile Communications, 2013. *Xperia Z: specifications*. [online] Available at: <http://www.sonymobile.com/global-en/products/phones/xperia-z/specifications/#tabs> [Accessed 21 October 2013].

SOS Publications Group, 201?. *Jargonbuster: technical terms explained*. [online] Available at: <http://www.soundonsound.com/information/Glossary.php> [Accessed 13 June 2013].

Steinberg Media Technologies, 2013a. *The Steinberg story*. [online] Available at: <http://www.steinberg.net/en/company/aboutsteinberg.html> [Accessed 20 December 2013].

Steinberg Media Technologies, 2013b. *Cubase* (7.0.2 trial version) [computer software] Available at: <http://www.steinberg.net/en/products/cubase/trial.html> [Accessed 19 November 2013].

Steinberg Media Technologies, 2013c. *Nuendo 5 in detail*. [online] Available at: <http://www.steinberg.net/en/products/nuendo/nuendo_6/whats_new/nuendo_5.html> [Accessed 27 June 2013].

Steinberg Media Technologies, 2013d. *Cubasis: a new world of recording*. [online] Available at: <http://www.steinberg.net/en/products/ios_apps/cubasis.html> [Accessed 24 July 2013].

Steinberg Media Technologies, 2014. *WaveLab*. [online] Available at: <http://www.steinberg.net/en/products/wavelab/start.html> [Accessed 8 January 2014].

Tammelin, O., 2013. *Buzz*. (build 1494) [computer software] Available at: <http://jeskola.net/buzz> [Accessed 15 September 2013].

Thalmic Labs, 2013. *MYO*. [online] Available at: <https://www.thalmic.com/myo/> [Accessed 13 June 2013].

The MusicRadar Team, 2012. *The 15 best DAW software apps in the world today*. [online] (10 October) Available at: <http://www.musicradar.com/tuition/tech/the-15-best-daw-software-apps-in-the-world-today-238905> [Accessed 13 June 2013].

Tidwell, J., 2006. *Designing interfaces*. Sebastopol, CA: O'Reilly Media.

Vintage Synth Explorer, 2011. *Electronic Music Studios (EMS) Synthi Sequencer 256*. [online] Available at: <http://www.vintagesynth.com/misc/synthi256.php> [Accessed 30 September 2013].

Vogelstein, F., 2013. And then Steve said, 'let there be an iPhone'. *The New York Times*, [online] 4 October. Available at: <http://www.nytimes.com/2013/10/06/magazine/and-then-steve-said-let-there-be-an-iphone.html?pagewanted=all&_r=3&> [Accessed 21 December 2013].

Walker, R., 2012. Freaks, geeks and Microsoft: how Kinect spawned a commercial ecosystem. *The New York Times*, [online] 31 May. Available at: <http://www.nytimes.com/2012/06/03/magazine/how-kinect-spawned-a-commercial-ecosystem.html> [Accessed 13 June 2013].

WaveMachine Labs, 2013. *Auria*. [online] Available at: <http://auriaapp.com/Products/auria> [Accessed 17 December 2013].

Waves, 2013a. *CLA-2A: user guide*. [pdf] Available at: <http://www.waves.com/1lib/pdf/plugins/cla-2a-compressor-limiter.pdf> [Accessed 2 July 2013].

Waves, 2013b. *PuigTec EQP-1A: user manual*. [pdf] Available at: <http://dl.owneriq.net/5/5f5042a1-9309-4f43-b96a-72144e269f93.pdf> [Accessed 2 July 2013].

White, P., 1994. Mixer anatomy: exploration. *Sound on Sound*, [online] August 1994. Available at: <http://www.soundonsound.com/sos/1994_articles/aug94/mixeranatomy.html> [Accessed 2 October 2013].

White, P., 2000. *The Sound On Sound book of desktop digital studio*. London: Sanctuary Publishing.

Wright, M, 2005. Open Sound Control: an enabling technology for musical networking. *Organised Sound*, [pdf] 10(3), pp. 193–200. New York: Cambridge University Press. Available at: <http://archive.cnmat.berkeley.edu/OpenSoundControl/publications/wright-organisedsound-OSC.pdf> [Accessed 8 January 2014].